7-7-2017

Instal·lació, configuració i validació d'un clúster BigData

TFG

Joan Josep Jiménez
4T GEI

ÍNDEX DE CONTINGUTS

1. INTRODUCCIÓ 5
1.1. Marc del projecte 5
1.2. Objectius del projecte 10
1.3. Planificació del projecte 11
1.3.1. Definició de tasques 11
1.3.2. Diagrama de Gantt 12
1.3.3. Anàlisi de costos 11
2. ESTAT DE L'ART 12
2.1. Clústers 12
2.2. BigData clusteritzat 15
2.3. Hadoop + HDFS + YARN 17
2.3.1. HDFS 17
2.3.2. YARN 18
2.3.3. Paradigma MapReduce 20
2.4. Serveis i aplicacions de Hadoop 21
2.4.1. Processament 21
2.4.2. Transferència de dades 22
2.4.3. Emmagatzemament 23
2.4.4. Gestió i configuració 23
2.5. Stacks de serveis/Frameworks 24
2.5.1. Stack de Hortonworks 25
2.5.2. Stack de Cloudera 26
2.5.3. Stack de MapR 27
3. INSTAL·LACIÓ I CONFIGURACIÓ CLÚSTER BIG DATA 29
3.1. Requisits Previs 29
3.2. Planificació Instal·lació 30
3.3. Definir arquitectura 32
3.4. Instal·lació y configuració sistema operatiu base 33
3.4.1. Instal·lació OS i paquets 33
3.4.2. Configuració clúster ssh passwordless 33
3.4.3. Configuració de la xarxa 34
3.4.4. Configuracions del sistema 35

3.5. Instal·lació Ambari 36
3.5.1. Servidor 36
3.5.2. Clients 36
3.6. Instal·lació Hadoop Stack i configuració dels serveis 37
3.6.1. Clúster base 37
3.6.2. HDFS 42
3.6.3. YARN MapReduce2 43
3.6.4. Spark2 44
3.7. Benchmarks BigData per a Hadoop/Spark 45
3.7.1. TestDFSIO 46
3.7.2. Terasort 47
3.7.3. HiBench 48
4. CONLUSIONS 50
4.1. Línies d'actuació futures 50
4.2. Opinió personal 51
5. BIBLIOGRAFÍA 52

ÍNDEX DE FIGURES

1.1 Evolució generació d'informació 5
1.2 Futur de la generació d'informació 6
1.3 3V BigData 6
1.4 MapReduce a Hadoop 8
1.5 Exemple d'ecosistema de Hadoop 9
1.6 Diagrama de Gant Projecte. 12
2.1 Arquitectura d'un clúster HPC 13
2.2 Arquitectura d'un clúster HA 13
2.3 Arquitectura de HTC BOINC 14
2.4 Arquitectura d'un clúster BigData 16
2.5 Serveis i aplicacions Hadoop 17
2.6 Mapa del HDFS 18
2.7 Funcionament recursos YARN 19
2.8 MR1 vs MR2 19
2.9 Aplicació MapReduce a Hadoop 20
2.10 Exemple MapReduce 20
2.11 HDP Stack [36] 25
2.12 Cloudera Stack [39] 26
2.13 MapR Stack [40] 27
2.14 Comparativa dels stacks [41] 28
3.1 Nodes "enrackats" 30
3.2 Xarxa del clúster 31
3.3 Distribució dels processos gestors del clúster 32
3.4 Directoris on es muntaran els dos HDD 33
3.5 Fitxer /etc/hosts de les màquines 34
3.6 Fitxer /etc/fstab de les màquines 35
3.7 Pas 1 37
3.8 Pas 2 37
3.9 Pas 3 38
3.10 Pas 4 39
3.11 Pas 5 39
3.12 Pas 6 40
3.13 Pas 7 40
3.14 Pas 8 41
3.15 Pas 9 41
3.16 Distribució de clients i serveis HDFS 42
3.17 Distribució de clients i serveis YARN 43
3.18 Configuració principal YARN 43
3.19 Distribució de clients i serveis de Spark i les dependències 44
3.20 Configuració contenidors TEZ 44
3.21 Permisos carpeta /benchmarks en el sistema HDFS 45
3.22 Executables de testing del sistema 45
3.23 Test d'escriptura del sistema HDFS 46
3.24 Test de lectura del sistema HDFS 46
3.25 Resultat Teragen 47
3.26 Resultat Terasort 47
3.27 Resultat Teravalidate 47
3.28 Resultat testos HiBench 49

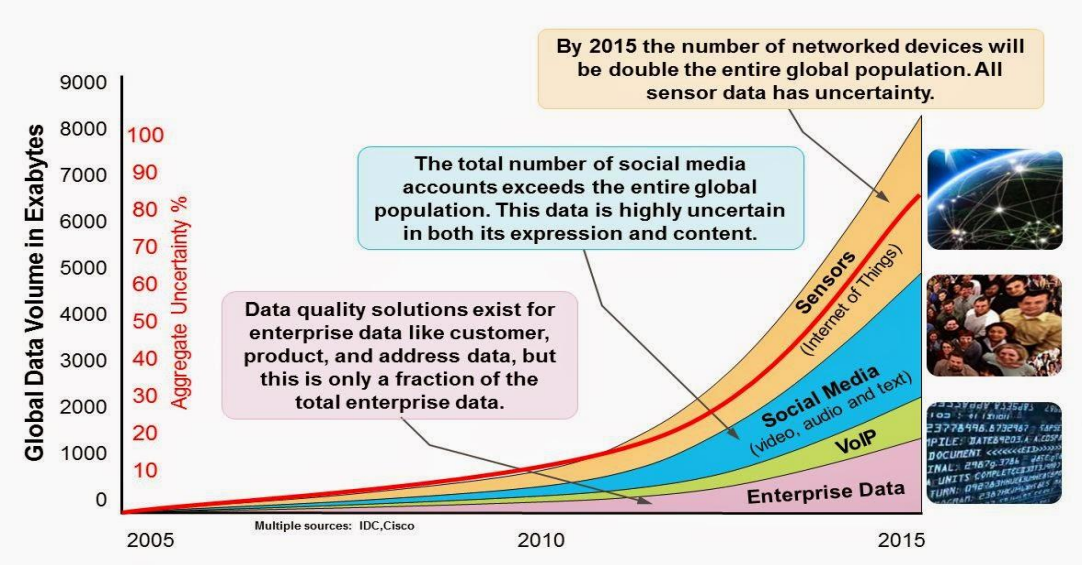
3.29 GRÀFICA DE TEMPS DELS TESTOS HIBENCH 49

1. INTRODUCCIÓ

1.1. Marc del projecte

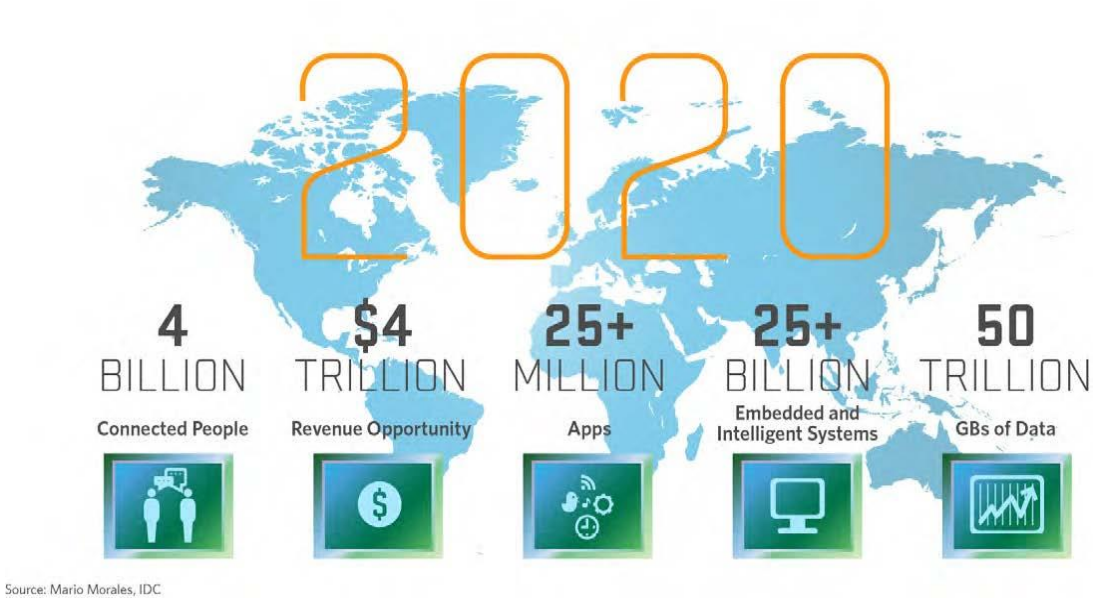
BigData [1] o dades massives, fa referència a una nova disciplina de tractament de grans volums d'informació, dit de forma més tècnica, és un conjunt de tècniques que permeten processar i treballar volums immensos de dades d'una forma més senzilla i adequada.

L'arribada de la "internet de les coses" ha generat una crescuda exponencial de la informació generada i, si ho sumem al creixement constant de les xarxes socials juntament amb la incertesa de la quantitat de dades que es poden generar en aquí, s'evidencia un ràpid increment del volum global d'informació. Si el 2010 els dos camps agrupaven uns 500 Exabytes, el 2015, tal com es veu en la figura 1.1, formen la gran majoria del volum de dades amb 6000 Exabytes, un volum 12 vegades més elevat en 5 anys aproximadament. [2]



1.1 Evolució generació d'informació

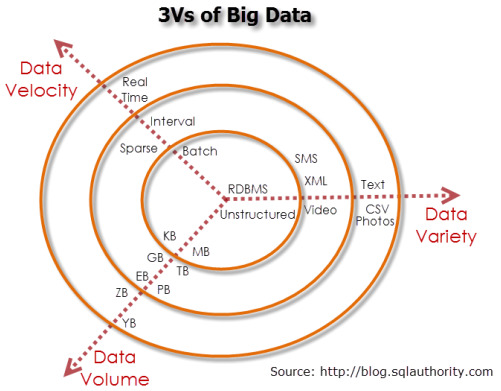
El futur es presenta encara amb més volum de dades, com mostra la Figura 1.2 gràcies als 4 bilions de persones connectades amb més de 25 milions d'aplicacions i més de 25 bilions de sistemes incrustats i intel·ligents en conjunt es generaran uns 50 trilions de GB d'informació. Amb aquestes expectatives de futur i en ple procés de creixement, per tractar i analitzar tota aquesta quantitat d'informació es necessita nous paradigmes i sistemes enfocats al processament massiu de dades. Aquesta és la raó, per la que el BigData és un focus d'inversió de futur per a grans empreses com Google, Microsoft, Intel, Amazon, etc. [3]



1.2 Futur de la generació d'informació

Els conjunts de dades considerats Big Bata solen complir la regla de les 3V [4]:

- Volum, actualment ja de diversos Petabytes.
- Velocitat, per aconseguir un processament en temps real o *streaming.*
- Varietat, les dades poden presentar múltiples formats.



1.3 3V BigData

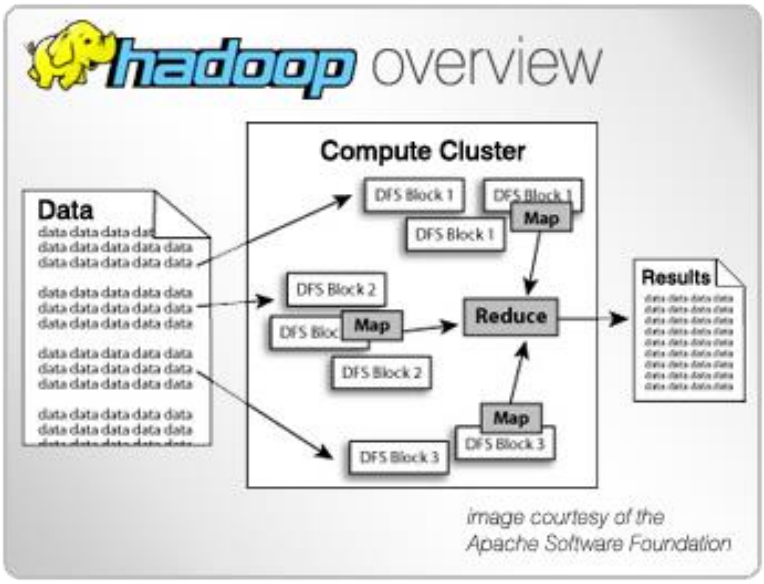
Recentment s'han afegit 2 Vs [5] addicionals a la definició anterior:

- Veracitat, donat l'elevat volum d'informació és impossible garantir la veracitat de totes les dades. Per definició les dades BigData són incertes, algunes vegades estan incompletes i contenen inconsistències. Les aplicacions BigData han de ser capaces de gestionar aquestes ambigüitats de la informació.
- Valor, és molt costós desplegar una infraestructura BigData per a poder emmagatzemar i processar aquest volum de dades. Per tant, aquest cost solament té sentit si es pot obtenir un valor afegit o benefici del processament de la informació.

Seguint les necessitats exposades anteriorment, processar les quantitats ingents de dades passa obligatòriament per la utilització d'infraestructures amb grans capacitats de tractament de dades, a ser possible, en temps real. Un únic ordinador no té suficients recursos per poder fer-se'n càrrec, per això, es fa necessari la utilització de clústers d'ordinadors. Un clúster [6] no és res més que un conjunt de màquines treballant coordinadament per generar una "supermàquina" amb una gran capacitat de processament. Tot i això, els clústers HPC (High Performance Computing) tradicionals no estan dissenyats per encarregar-se d'aquesta feina. Els clústers HPC tenen per objectiu proporcionar grans quantitats de còmput per a aplicacions paral·leles. Mentre que les aplicacions BigData necessiten una alta capacitat de disc per emmagatzemar les dades i un gran ample de banda d'E/S (Entrada/Sortida) per a processar les dades i generar els resultats. Per això fa falta un clúster que ofereixi les capacitats de computació i E/S necessàries per processar BigData.

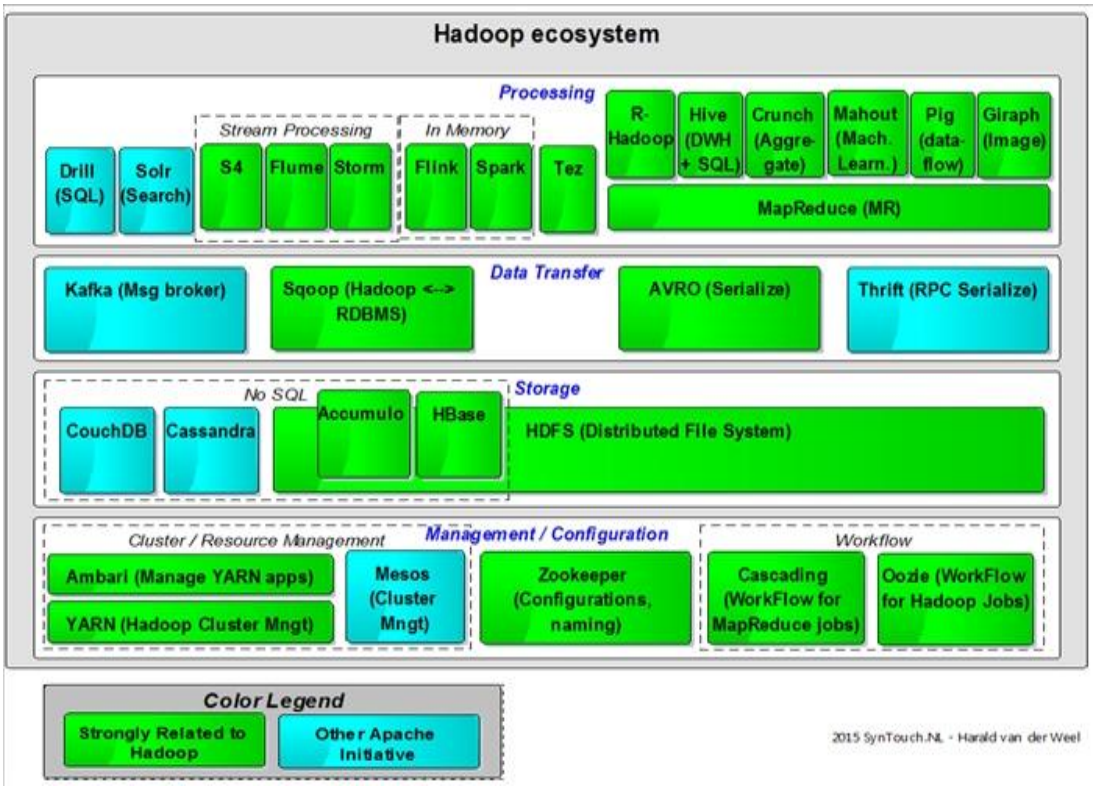
Un clúster BigData es caracteritzarà per tenir una gran capacitat d'emmagatzemament d'informació amb les prestacions necessàries per a processar-la, juntament amb l'escalabilitat inherent en un clúster i la tolerància a fallades per assegurar que la informació no es perdi.

Dins del món del BigData l'estàndard és el *framework* Apache Hadoop. [7] Començant el 10 de desembre del 2006 i basant-se amb documents de Google sobre el *MapReduce* [8] i *Google File Sistem* (GFS) [9], Hadoop basa la seva funcionalitat en un sistema de fitxers distribuït HDFS i aplicacions que utilitzen el paradigma *MapReduce* tal com es mostra en la figura 1.4. [10]



1.4 MapReduce a Hadoop

L'entorn Hadoop te una gran varietat d'eines i serveis que ofereix tant per la part *open source* com pel món privat. Tot i això fa que el sistema sigui molt complex de configurar per la seva diversitat funcional figura 1.5, tenint per exemple un sistema d'emmagatzemament No-SQL com HBase, YARN com a gestor de recursos o Spark [11] com a eina que permet execucions en memòria d'aplicacions que utilitzin el paradigma *MapReduce* fent-les molt més ràpides en comparació amb Hadoop. [12]



1.5 Exemple d'ecosistema de Hadoop

El fet que l'entorn Hadoop sigui tan flexible en serveis i aplicacions juntament amb una àmplia comunitat i documentació de què disposa fa que aquest entorn tingui la iniciativa i marquí la majoria del camí pels clústers BigData, tal és així que companyies com Intel, Microsoft o Hortonworks creen a partir de Hadoop, els seus propis entorns de desenvolupament. Per tant l'entorn BigData Hadoop és el més indicat per realitzar els objectius d'aquest treball.

Aquest projecte consisteix en la realització del desplegament i configuració d'un clúster híbrid d'investigació BigData basat en Hadoop que permeti l'execució eficient d'aplicacions BigData en un entorn multiusuari. Aquest tipus d'instal·lacions d'investigació es caracteritzen per exigir un entorn d'execució controlat on es puguin replicar els diferents experiments. Això obliga a la utilització de gestors de treballs tipus YARN o SGE, que permetin la compartició del clúster per diferents aplicacions, però garantint la utilització de recursos en exclusiva per cada una d'elles. També seria interessant que aquest clúster permetés una utilització híbrida per part d'aplicacions BigData i aplicacions HPC. Això implica que es necessita instal·lar ambdós *frameworks* (Hadoop i MPI) i aconseguir que puguin coexistir simultàniament, sense molestar-se entre si.

1.2. Objectius del projecte

Aquest clúster a l'estar destinat a la investigació, haurà de comptar amb un nivell de seguretat suficient per evitar conflictes entre els usuaris. S'ha optat per fer-lo híbrid per les restriccions monetàries de no poder crear dos clústers, un BigData i un altre per MPI.

- Implementar i configurar un clúster híbrid BigData basat en Hadoop amb MPI.
- Tots els serveis i aplicacions de Hadoop s'hauran de poder executar de forma segura per evitar possibles problemes originats en la manca de control d'usuaris.
- Garantir el correcte funcionament dels serveis i aplicacions instal·lats en l'entorn.

1.3. Planificació del projecte

Per poder realitzar el projecte s'han definit quatre grans tasques: La primera està relacionada amb els estudis de les diverses tecnologies necessàries per al projecte, és a dir, Hadoop i tot el seu ecosistema de serveis i eines (Ambari, Spark, etc.). A continuació caldrà realitzar la instal·lació i desplegament del clúster de Bigdata, per acabar amb totes les tasques de validació que ens permetrà comprovar que tots els components del clúster funcionen correctament. Finalment l'escriptura de la documentació associada amb el projecte es realitzarà de forma contínua al llarg del mateix.

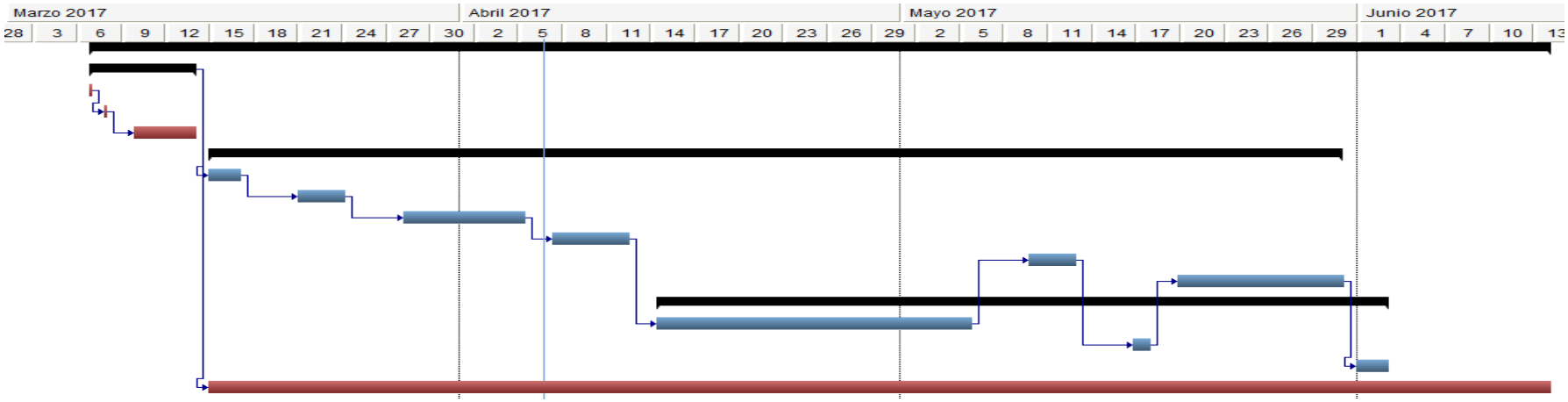
1.3.1. Definició de tasques

Les tasques a realitzar consten de:

- Anàlisi de les tecnologies i solucions requerides pel projecte
  - Anàlisi de les tecnologies actuals BigData. 1 dia
  - Cercar les diferents opcions de seguretat que ofereixi el sistema BigData escollit. 1 dia
  - Cercar i estudiar solucions per crear el clúster híbrid amb MPI. 2 dies
- Instal·lació i configuració del clúster.
  - Instal·lació i configuració del sistema operatiu en els nodes del clúster. 2 dies
  - Instal·lació i configuració dels serveis Ambari al node *master* i nodes *worker*. 3 dies
  - Instal·lació i configuració de les aplicacions i serveis BigData necessaris des del gestor Ambari. 5 dies
  - Aplicació de les polítiques de seguretat adients en les aplicacions i serveis BigData instal·lats. 3 dies
  - Instal·lació dels serveis MPI+SGE al clúster. 3 dies
  - Configuració del *script* que permeti apagar i encendre els dos models de clúster. 5 dies
- Comprovació del correcte funcionament del clúster
  - Comprovació del correcte desplegament de les aplicacions i serveis instal·lats. (*Testing*) 10 dies
  - Comprovació del desplegament de MPI+SGE. 2 dies
  - Comprovació del correcte funcionament del *script*. 2 dies
  - Aquestes tasques es realitzaran modularment, és a dir, s'executarà cada una un cop acabada la part del projecte que la comprèn.
- Documentació del projecte en la memòria.
  - Aquesta tasca s'anirà completant per fases a mesura que vagi avançant el projecte. 30 dies.

1.3.2. Diagrama de Gantt

Nombre	Duración	Esfuerzo	Inicio	Fin	Predecesoras
TFG	**43d?**	**312h**	**06/03/2017**	**12/06/2017**	
Anàlisi de les tecnologies i solucions requerides pel project	**4d**	**16h**	**06/03/2017**	**13/03/2017**	
Anàlisi de les tecnologies actuals BigData	1d	4h	06/03/2017	06/03/2017	
Cercar les diferents opcions de seguretat que ofereixi el s	1d	4h	07/03/2017	07/03/2017	3
Cercar i estudiar solucions per crear el clúster híbrid amb	2d	8h	09/03/2017	13/03/2017	4
Instal·lació i configuració del clúster	**31d**	**84h**	**14/03/2017**	**23/05/2017**	
Instal·lació i configuració del sistema operatiu en els node	2d	8h	14/03/2017	16/03/2017	2
Instal·lació i configuració dels serveis Ambari al node mas	3d	12h	20/03/2017	23/03/2017	7
Instal·lació i configuració de les aplicacions i serveis BigD	5d	20h	27/03/2017	04/04/2017	8
Aplicació de les polítiques de seguretat adients en les apl	3d	12h	06/04/2017	11/04/2017	9
Instal·lació del serveis MPI SGE al clúster	3d	12h	08/05/2017	11/05/2017	14
Configuració del script que permeti apagar i encendre els	5d	20h	15/05/2017	23/05/2017	15
Comprovació del correcte funcionament del clúster	**20d**	**56h**	**13/04/2017**	**29/05/2017**	
Comprovació del correcte desplegament de les aplicacion	10d	40h	13/04/2017	04/05/2017	10
Comprovació del desplegament de MPI SGE	2d	8h	15/05/2017	16/05/2017	11
Comprovació del correcte funcionament del script	2d	8h	25/05/2017	29/05/2017	12
Documentació del projecte en la memòria	39d	156h	14/03/2017	12/06/2017	2



1.6 Diagrama de Gant Projecte.

1.3.3. Anàlisi de costos

El projecte pot constar de la participació d'un analista i d'un programador. L'analista seria l'encarregat de fer tota la recerca, documentació final i part del *testing* per comprovar-ne la qualitat. El programador s'encarregaria de tota la resta de la instal·lació, configuració i documentació inicial.

A partir del cost per al client de les hores dels dos, s'ha arribat a què l'analista costaria uns 50 €/h i el programador uns 20 €/h.

La següent taula mostra les hores i el cost d'aquestes per a dur a terme el projecte.

	Hores	Cost
Analista	100	5000 €
Programador	212	4240 €
TOTAL	**312**	**9240 €**

2. ESTAT DE L'ART

2.1. Clústers

Seguint l'esmentat en l'apartat anterior es necessitarà un clúster. [6]

Un clúster consta de diversos components:

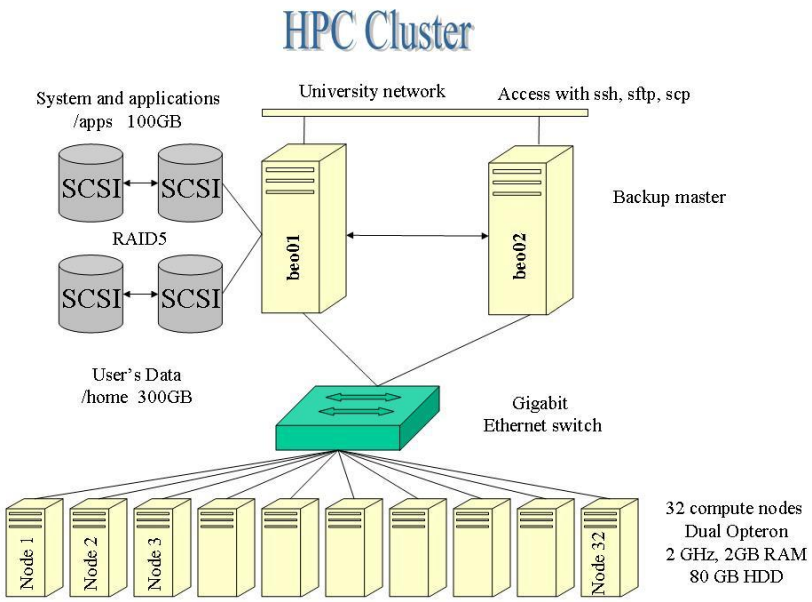
Els **nodes** són la massa de treball del clúster, solen ser tots d'ordinadors amb característiques similars (homogenis o heterogenis) per mantenir una correcta eficiència.

Els ***master (front-end)*** o nodes dedicats corresponen a nodes que poden ser més potents que la resta, són els encarregats de la gestió, supervisió i manteniment del funcionament dels serveis del clúster.

La **xarxa** és l'encarregada d'interconnectar tots els components, generalment és una xarxa d'alta velocitat com Gigabit per tal de reduir al mínim la latència entre nodes.

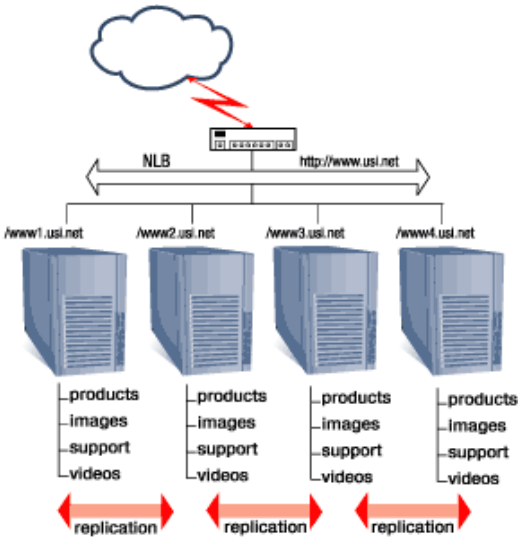
Actualment existeixen tres tipus diferents de clústers:

Alt rendiment (HPC *High Performance Computing*) són clústers que es caracteritzen per disposar de grans quantitats de memòria i capacitat de processament, s'utilitzen per realitzar tasques que requereixen molts recursos. Generalment distribuïts com s'observa en la figura 2.1 Arquitectura d'un clúster HPC de tal forma que tenim una gran quantitat d'elements de treball interconnectats i uns altres de gestió. Per a poder garantir el correcte ús de la capacitat de càlculs d'aquests clústers, és necessari que els problemes siguin paralelitzables, ja que el mètode amb què els clústers agilitzen el processament és dividir el problema en problemes més petits i calcular-los en els nodes. Perquè els problemes siguin paralelitzables han de fer ús de biblioteques especials com MPI (Message Passage Interface). Un exemple d'aquest tipus de clústers són els que utilitzen SGE (*Sun Grid Engine*) [14] + MPI [15], el qual funcional amb SGE que és un sistema de cues dissenyat per permetre executar diverses tasques en diferents màquines d'un clúster. Un altre exemple és el Marenostrum del BSC.



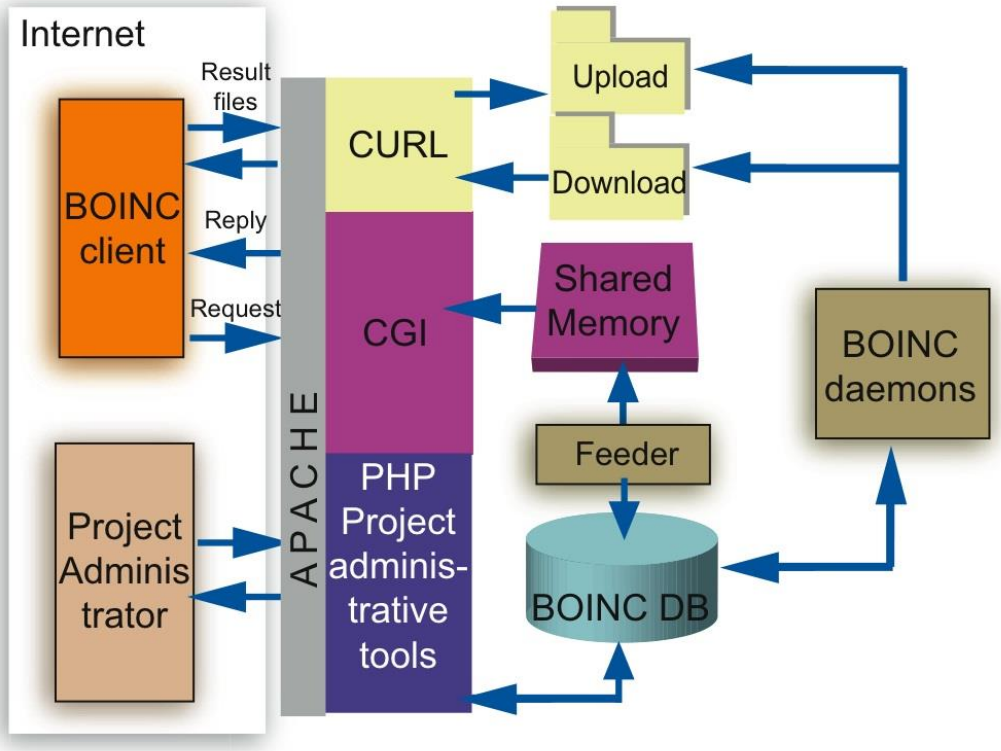
2.1 Arquitectura d'un clúster HPC

Alta disponibilitat (HA *High Availability*) [13] són clústers destinats a proveir de disponibilitat i fiabilitat als serveis que ofereixin. Es caracteritzen per mantenir una sèrie de serveis compartits i estar constantment monitorant entre si de tal forma que, d'haver-hi un error de hardware en algun node, el software és capaç d'executar els serveis en un altre node del conjunt (*failover*). Un cop el node que ha fallat es recupera, els serveis són migrats altre cop al node originari (*failback*), garantint així que en cas d'haver-hi errors d'infraestructura la percepció de l'error sigui mínima per l'usuari. Tot això és possible gràcies a l'ús de rèpliques (*mirrors*) distribuïdes entre els nodes com s'observa en la figura 2.2 per tal de tenir múltiples còpies sempre disponibles. S'utilitza molt en serveis populars d'internet com Facebook, Google, etc.



2.2 Arquitectura d'un clúster HA

Alta eficiència (HTC *High Throughput Computing*) és un concepte aplicat a clústers que basen el seu funcionament a poder executar la quantitat més gran de tasques en el menor temps possible. En essència aquests clústers són similars als HPC però el seu objectiu és diferent, mentre el HPC busca fer una operació el més ràpid possible, HTC busca obtenir el màxim d'operacions realitzades en un temps concret. Existeixen alguns *frameworks* HTC com HTCondor [16] per poder utilitzar el concepte HTC en un clúster o el projecte BOINC [17]. La figura 2.3 mostra un exemple de clúster de BOINC.



2.3 Arquitectura de HTC BOINC

2.2. BigData clusteritzat

La tecnologia BigData [1] com a tractament d'informació sol seguir tres fases:

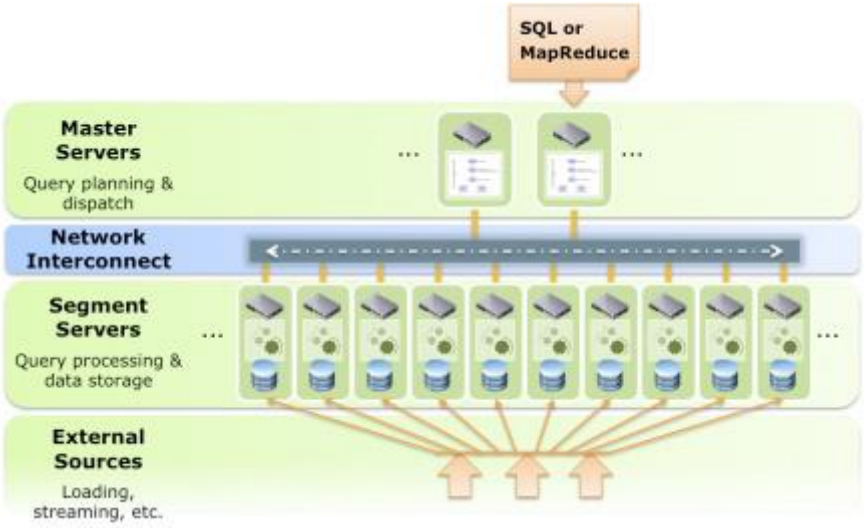
La **captura** és del procés en el qual s'obtindran totes les dades a processar, els orígens són variats, poden ser generades per les persones o empreses en la seva activitat informàtica, pe: *tweets*, moviments bancaris, compres, etc. O per institucions públiques com la Seguretat Social, en la que s'emmagatzemin dades biomèdiques o extractes de casos clínics. Aquest model veurà en el nou model de *Internet of Things* un gran nínxol de dades, ja que com s'exposava en la introducció, aquest és el model en el que es generen i es generaran les quantitats més grans de dades. Un exemple d'aquest nou model creixent són les *SmartCities* que veuen en el BigData la millor manera de processar la ingent quantitat d'informació pròpia que genera una ciutat a gestionar de forma digital.

La **transformació**, un cop establerts els orígens de les dades, s'utilitza el procés *Extract,_transform_and_load* (ETL) que consta d'agrupar, transformar i filtrar les dades per poder-les guardar en les pertinents bases de dades.

L'**emmagatzemament** en bases de dades que poden ser del tipus adient en cada cas sigui, SQL o NoSQL (Clau-Valor).

Un clúster per a processar BigData és un clúster enfocat al tractament massiu d'informació, per tant, tindrà un ampli sistema encarregat de l'emmagatzemament i processament de dades. Això s'aconsegueix a partir de la creació de nodes de dades i nodes de processament juntament amb un nou paradigma, *MapReduce*, pel processament d'aquestes dades de forma paral·lela/distribuïda de forma molt eficient. I que simplifica la implementació, escalabilitat i la gestió de la tolerància a fallades.

En la figura 2.4 s'observa que l'arquitectura és similar als clústers HPC, però la diferència resideix en l'enfocament del software que executarà el sistema, ja que els dos apliquen enfocaments diferents, com ja s'ha comentat, els HPC la seva finalitat és el procés massiu general sense ser òptim pel procés massiu de dades i el BigData resideix en processar, tractar i analitzar òptimament grans agrupacions de dades.



2.4 Arquitectura d'un clúster BigData

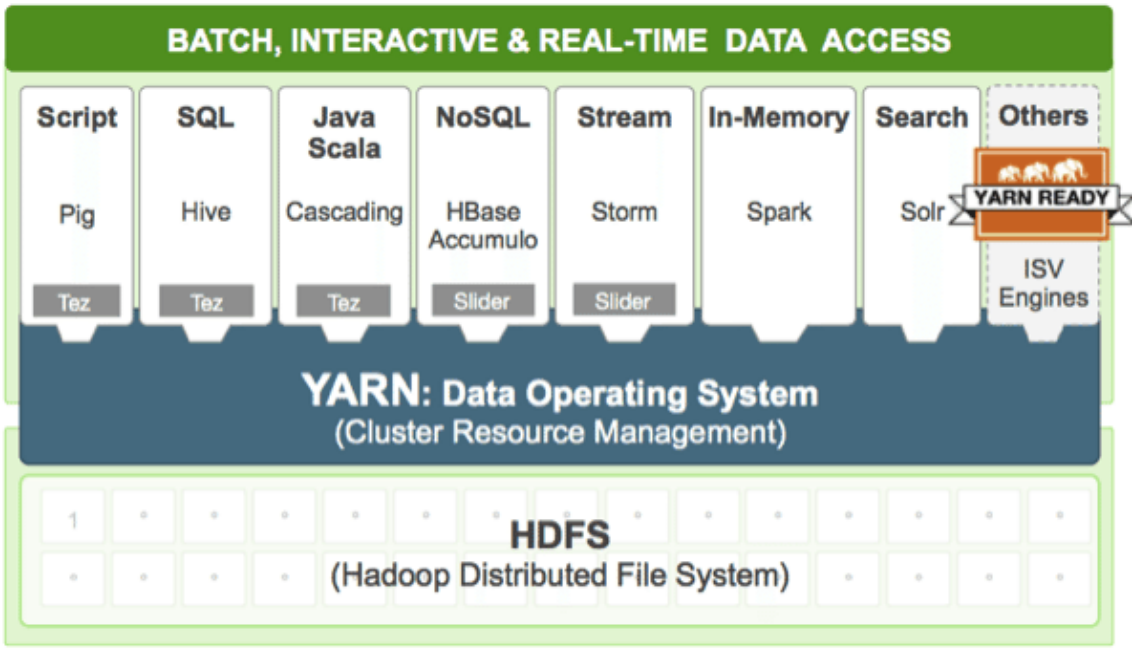
Un clúster BigData utilitzarà un sistema d'arxius distribuït a gran escala per tal de poder gestionar la immensa quantitat de dades a processar de forma segura i garantint-ne la disponibilitat. Aquest sistema serà similar als clústers HA, ja que garantirà una òptima gestió dels errors en els nodes mitjançant la replicació de les dades, fent-lo així, tolerant a fallades.

El clúster emprarà també una capa de software capaç d'abstraure els recursos del conglomerat de màquines, ja que, no sempre es disposarà de sistemes homogenis, per exemple, els nodes *master* requeriran més recursos per tal de poder gestionar òptimament tots els serveis de monitoratge i administració, els nodes de dades no requeriran massa recursos de còmput però si un ampli espai de disc, també els nodes de processament no tenen per què requerir massa espai de disc però, en canvi, si necessitaran una quantitat suficient de capacitat de processament per dur a terme les tasques designades.

Finalment gràcies al sistema distribuït, les tasques de migració/replicació de tasques/dades es tornen altament eficients i senzilles, aportant robustesa a les tasques que hagi d'executar el clúster.

2.3. Hadoop + HDFS + YARN

Dins del món del BigData existeixen varietat de sistemes i mecanismes, el més conegut és el *framework* Apache Hadoop. Hadoop [7] basa la seva funcionalitat en un sistema de fitxers distribuït HDFS i aplicacions que utilitzen el paradigma *MapReduce* com s'aprecia a la figura 2.5.

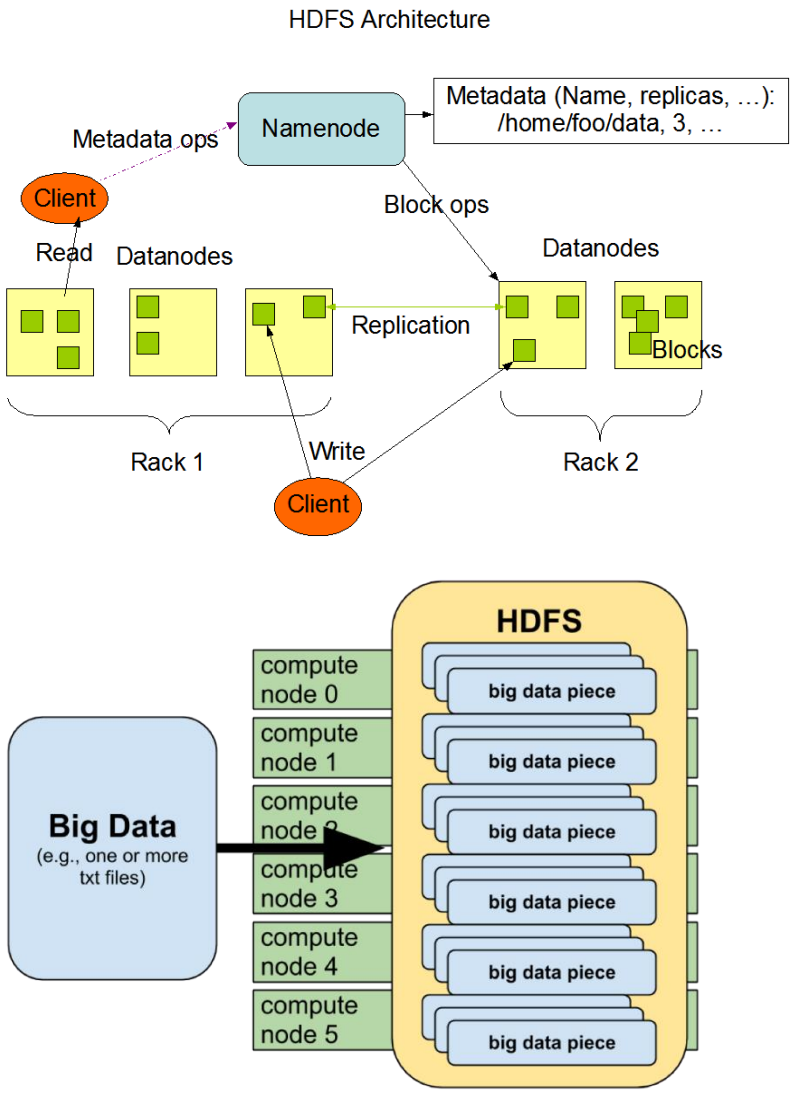


2.5 Serveis i aplicacions Hadoop

2.3.1. HDFS

El sistema HDFS (*Hadoop Distributed File System*) [18] és un sistema d'arxius distribuït escrit en Java inclòs en el *framework* i és la base de Hadoop. HDFS basa el seu funcionament a partir les dades en blocs de X MB (l'ideal és 64 MB) i escriure-les entre els diferents nodes com es mostra en la figura X amb una replicació general de 3, és a dir, cada bloc estarà com a mínim en 3 nodes diferents per tal de garantir-ne una correcta disponibilitat tot i que, HDFS no proporciona les propietats dels clústers HA.

En la figura 2.6 es representa com els nodes poden comunicar-se entre si per balancejar les dades, moure copies i conserva la replicació, tot això permet evitar la necessitat de tenir un node extra de dades per cada node ni tampoc es fa necessari un RAID de dades.



2.6 Mapa del HDFS

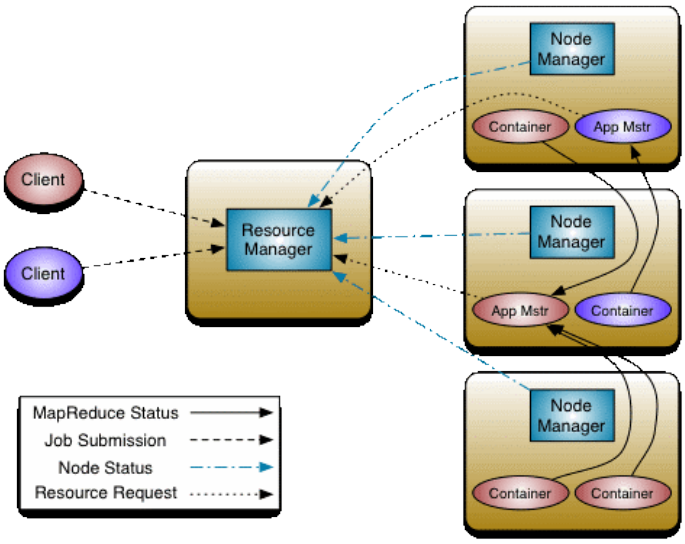
2.3.2. YARN

Per gestionar els recursos del clúster per tal de coordinar i repartir tasques *MapReduce* s'utilitza un gestor de recursos a partir de cues, el més conegut i utilitzat és el Apache Hadoop YARN [19].

La idea fonamental de YARN és crear cues de tasques i monitorar-les amb processos separats a partir de diferents components:

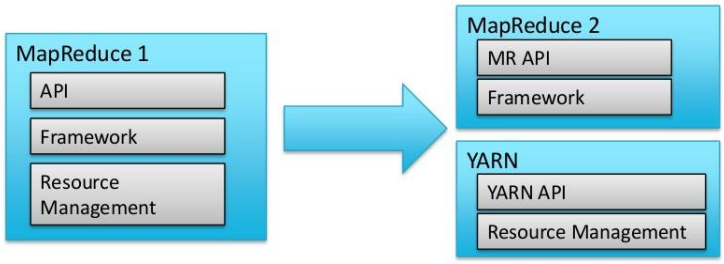
- El ***ResourceManager*** és l'encarregat que gestiona els recursos de totes les aplicacions del sistema.
- El ***NodeManager*** és el component que està en cada node de treball encarregat de gestionar i monitorar els contenidors de recursos (CPU, memòria, disc, xarxa) i comunicar-ho al *ResuourceManager*.

Cada aplicació haurà d'utilitzar una llibreria anomenada *ApplicationMaster* que seguint el mostrat en la figura 2.7, és l'encarregada de negociar els recursos amb el *ResourceManager* i treballar amb el *NodeManager* a través dels contenidors per executar i controlar les tasques de l'aplicació.



2.7 Funcionament recursos YARN

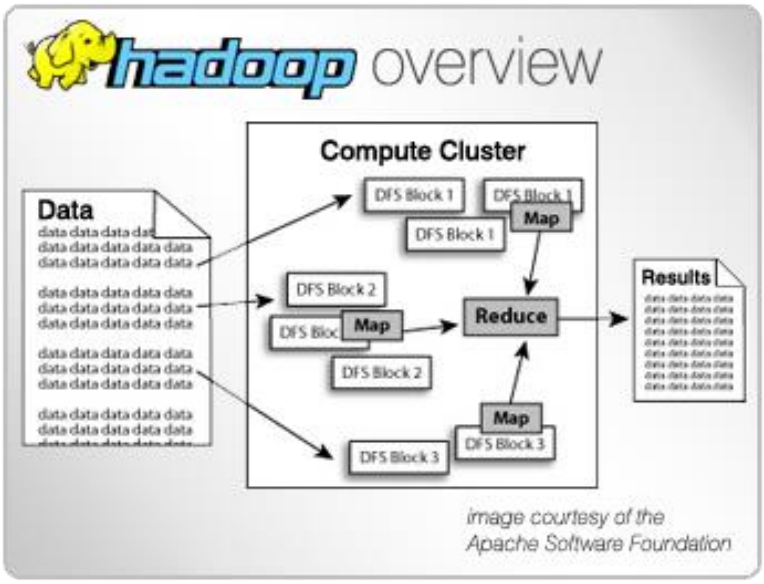
L'aplicació del gestor YARN va incentivar la creació de *MapReduce2*. [20] Com mostra la figura 2.8 bàsicament la diferència és que en comptes de ser el mateix *MapReduce* l'encarregat de gestionar els recursos, ho fa YARN.



2.8 MR1 vs MR2

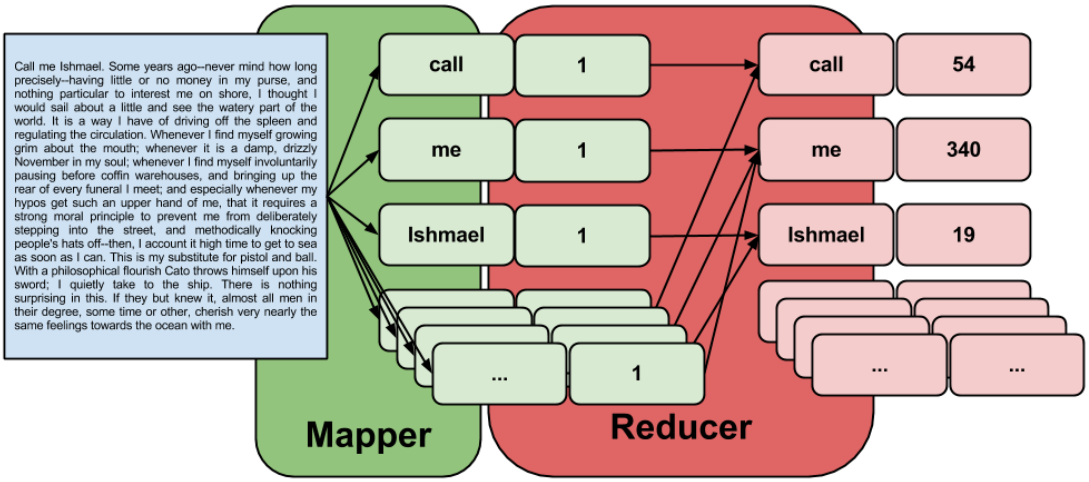
2.3.3. Paradigma MapReduce

Tal com s'il·lustra a la figura 2.9, les aplicacions *MapReduce* [10] basen el seu funcionament en quatre passos, recol·lecció, filtrat i *mapeig*, aplicació d'un procés *reduce* i finalment mostrar resultats sobre un massiu de dades.



2.9 Aplicació MapReduce a Hadoop

Un exemple de *MapReduce* és el d'un comptador de paraules, en el que se li aplicarien les següents accions com mostra la figura 2.10.



2.10 Exemple MapReduce

El procés de **recol·lecció** pot ser diferent depenent de l'origen de les dades, tant pot ser un fitxer de text, com una cadena constant de dades entrants (*stream*). En el nostre exemple podria ser un article d'internet.

La fase de ***mapeig* [Map],** basa la seva idea a realitzar un determinat processament/tractament de les dades d'entrada. Per exemple les dades que arriben de la recol·lecció no tenen per què ser totes útils, pot haver-hi "palla" a eliminar i per això s'apliquen filtres. Els map escriuen la informació processada per tal de ser "*mapejada"* seguint unes regles descrites que les agruparà seguint un esquema Clau-Valor (K,V) per a ser processades de forma òptima pel sistema. En l'exemple s'eliminarien tots els espais o paraules que no es volguessin en el mapa i es "*mapejaren*" seguint el sistema (Paraula, Núm. de cops que apareix).

L'etapa de **reducció/agregació [Reduce]** és l'essència de tot el paradigma. Per tal de dur-se a terme s'haurà d'haver-lo proveït d'un algoritme o funció que processi cada parella Clau-Valor del mapa obtingut en la fase anterior i que, en el mateix, consti com s'han d'agrupar/reduir els resultats obtenint un mapa de valors nou i, més petit que el de la fase anterior. El procés aplicat al mapa de l'article agafaria cada parella (Paraula, Núm. repeticions) fent que, cada vegada que es trobi una paraula nova s'afegeixi al mapa resultat tal que (Paraula, 1), i de ser una paraula ja existent al mapa resultat, s'incrementaria en 1 el Núm. de repeticions (Paraula, Núm. repeticions actuals + 1).

Finalment a l'hora de **mostrar resultats**, s'agafarà el mapa obtingut en la fase de *reduce* i s'imprimiran els resultats seguint les ordres de l'usuari, sigui veure-ho per pantalla o guardar-ho en un fitxer de nou al sistema de fitxers. En el nostre cas podria ser una senzilla impressió de la llista de paraules més el nombre de repeticions contades.

2.4. Serveis i aplicacions de Hadoop

A partir de la Figura 1.5, observem que a part dels elements comentats anteriorment, HDFS, YARN i *MapReduce*, Hadoop disposa d'altres eines [21]:

2.4.1. Processament

- **Apache Drill:** És una rèplica de Google Dremel que suporta diferents tipus de dades NoSQL i sistemes de fitxers. S'utilitza per a poder proveir d'escalabilitat per tal de processar dades eficientment de forma comuna independentment del seu origen amb una sola consulta.
- **Apache Solr:** És un servei encarregat de cerca i indexar en l'ecosistema Hadoop. Està basat en Apache Lucene.
- **Apache Flume:** És un servei distribuït, confiable i disponible per a recopilar, agregar i moure eficientment grans quantitats de dades de

log. Té una arquitectura simple i flexible basada en fluxos de dades en *streaming*.

- **Apache Storm:** És un sistema de computació en temps real lliure i obert. Storm facilita el processament fiable de fluxos de dades sense límits utilitzant una topologia DAG (*Directed Acyclic Graph*) entre els orígens de les dades i el destí. [22]
- **Apache Flink:** És un *framework* programat en Java i Scala encarregat de proveir un motor de baixa latència i d'alta sortida (HTC) per a processar conjunts de dades finites o infinites.
- **Apache Spark:** És un *framework* pensat per executar anàlisi de dades distribuïdes en un entorn de computació distribuïda. Executa processos en la memòria per tal d'augmentar la velocitat d'execució fins a 100 vegades respecte a una execució sobre Hadoop normal.
- **Apache Tez:** És un complement pensat per permetre a aplicacions que utilitzen un complex DAG processar dades. Per exemple permetent Hive i Pig executar complexos DAG, Tez es pot fer servir per a processar les dades que abans requerien diferents tasques *MapReduce* requerint-ne només una. [24]
- **Apache Hive:** És una eina creada per Facebook per aconseguir obtenir un entorn similar a SQL dins el sistema Hadoop.
- **Apache Crunch:** És una llibreria de Java que proporciona un marc per a escriure, provar i executar *pipelines MapReduce*. El seu objectiu és fer que els *pipelines* compostos per funcions definides per l'usuari siguin senzills d'escriure, senzilles de provar i eficients. [25]
- **Apache Mahout:** És una extensió per a crear aplicacions *Machine learning* escalables dins l'entorn Hadoop.
- **Apache Pig:** És un complement que permet executar scripts sobre Hadoop per tal d'executar aplicacions *MapReduce* amb un codi més senzill. Aproximadament 10 línies de Pig equivalen a 200 línies de *MapReduce* en Java.
- **Apache Giraph:** És un *framework* per el processament iteratiu de grafs sobre de Hadoop i que proporciona una alta escalabilitat. [26]

2.4.2. Transferència de dades

- **Apache Kafka:** És un projecte encarregat de proporcionar una plataforma unificada, d'alt rendiment i de baixa latència per a la manipulació en temps real de fonts de dades. Similar a una cua de missatges sota el patró publicació-subscripció. [27]
- **Apache Sqoop:** És una eina similar a Flume però en aquest cas serveix per importar o exportar dades estructurades entre Hadoop i bases de dades estructurades, com les BD relacionals.

- **Apache AVRO:** És una crida dins de l'entorn Hadoop que fa servir JSON per a definir tipus de dades i protocols, permet serialitzar dades en un format binari compacte. Dins l'entorn es fa servir per a comunicació entre nodes i entre clients de serveis de Hadoop. [28]
- **Apache Thrift:** És una interfície que es fa servir per tal de definir i crear serveis per diferents llenguatges. [29]

2.4.3. Emmagatzemament

- **Apache CouchDB:** És un gestor de base de dades a través de web pensat per a fer-ne fàcil l'ús. És una base de dades NoSQL que utilitza JSON per emmagatzemar dades, JavaScript com a llenguatge de consulta a partir d'operacions *MapReduce* i HTTP com a API. [30]
- **Apache Cassandra:** És una base de dades NoSQL distribuïda que permet grans volums de dades de forma distribuïda amb un gran rendiment. [31]
- **Apache Accumulo:** És un software que proveeix un emmagatzemament Clau-Valor ordenat, basat en la tecnologia BigTable [32] de Google. [33]
- **Apache HBase:** És similar a Hive, però en aquest cas és una base de dades NoSQL dins l'entorn Hadoop.

2.4.4. Gestió i configuració

- **Apache Ambari:** És un gestor encarregat de simplificar i automatitzar tot el procés d'instal·lació, gestió i mantenir un clúster basat en Hadoop. Permet mantenir un control coordinat sobre tots els aspectes rellevants dels serveis del clúster, a part de poder extreure'n estadístiques útils de l'ús dels recursos, etc.
- **Apache Mesos:** És un gestor dels recursos de clústers BigData que permet la compartició dels recursos de forma precisa millorant la utilització del clúster. [34]
- **Apache Zookeeper:** És el coordinador de qualsevol treball dins de Hadoop, coordina diversos serveis dins l'entorn. Gràcies a aquesta eina s'estalvia molts problemes de sincronització, configuració, manteniment, agrupació i nomenament.
- **Cascading:** És un software per afegir una capa d'abstracció sobre Hadoop i Flink. Es fa servir per crear i executar complexos processaments de dades ocultant la complexitat de les tasques *MapReduce*. [35]
- **Apache Oozie:** És un sistema basat en servidor per a la planificació de fluxos de treballs per a gestionar els treballs de Hadoop. Els fluxos de treball a Oozie es defineixen com una col·lecció de nodes de flux, de control i d'acció en un DAG.

- **HUE:** HUE (Hadoop User Experience) és una interfície web de codi obert que suporta Apache Hadoop i el seu ecosistema. Proporciona un conjunt d'aplicacions que permeten interactuar amb un clúster Hadoop. Les aplicacions HUE permeten navegar per HDFS, administrar Hive i executar consultes Hive i Impala, comandes HBase i Sqoop, scripts Pig, treballs MapReduce i fluxos de treball de Oozie.

2.5.Stacks de serveis/Frameworks

Per a determinar quin proveïdor de Hadoop és correcte per la instal·lació és necessari contemplar diverses característiques clau. Aquestes inclouen el model de desplegament, les funcionalitats empresarials, la seguretat i la protecció de dades i els serveis que suporta.

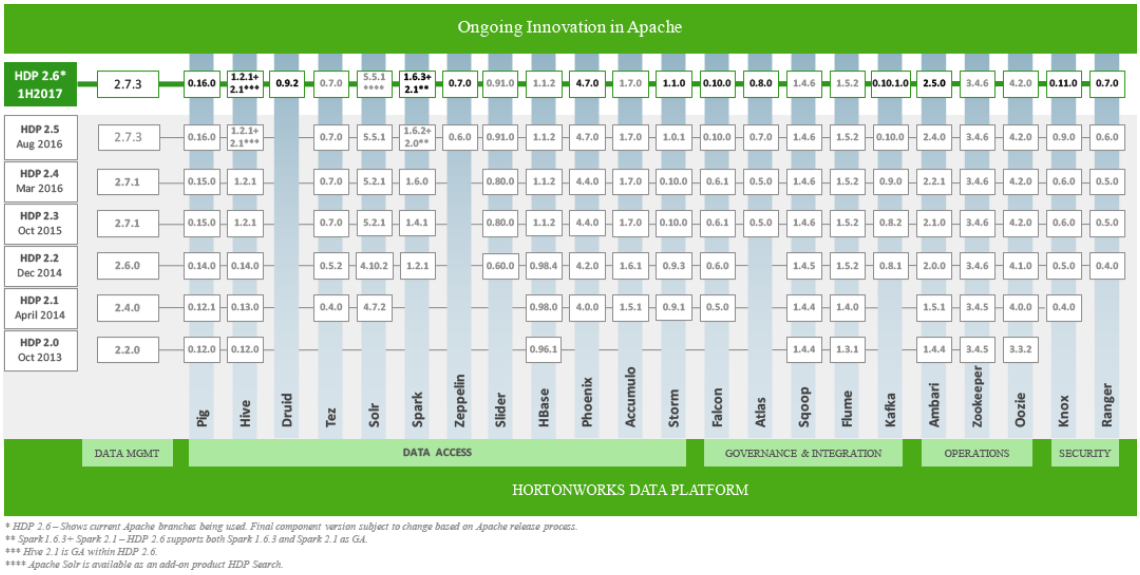
Cal tenir en compte que mentre l'entorn de Hadoop està pensat per suportar emmagatzemament d'informació escalable i computació distribuïda d'alt rendiment, el rendiment final pot variar a causa de diferents motius relacionats amb la implementació del software.

Mentre que els grans distribuïdors de Hadoop (Cloudera, Hortonworks. IBM i MapR) ofereixen desplegaments basats en el núvol, no estan limitats a aquest model. Permeten als usuaris descarregar les distribucions que puguin ser desplegades en local o en clouds privats en diversos servidors, incloent-hi sistemes Linux i Windows. Cloudera, Hortonworks i MapR també proveeixen de versions sandbox que poden ser executades en màquines virtuals.

2.5.1. Stack de Hortonworks

Hortonworks és una nova empresa en el mercat fundada el 2011 com una companyia independent amb orígens a Yahoo. Hortonworks està centrada a proveir una solució *open source* fet que la fa l'única companyia en fer-ho. L'oferta de Hortonworks resideix en el *Hortonworks Data Platform* (HDP) construït fent servir Apache Hadoop. Gràcies al fet que és una plataforma de codi obert permet que sigui molt més fàcil d'implementar millores i novetats. Empreses com Ebay, Bloomberg, Spotify i Sambung utilitzen Hortonworks. [36]

L'empresa està vinculada en un projecte per a gestionar les dades a Hadoop, amb un enfocament inicial en el nou projecte d'Apache anomenat Atlas per a la gestió de metadades compartides, classificació de dades, auditoria i gestió de la seguretat i polítiques per a la protecció de dades. A la vegada treballa per integrar Atlas amb Ranger, una eina de seguretat de codi obert per aplicar les polítiques d'accés a les dades.



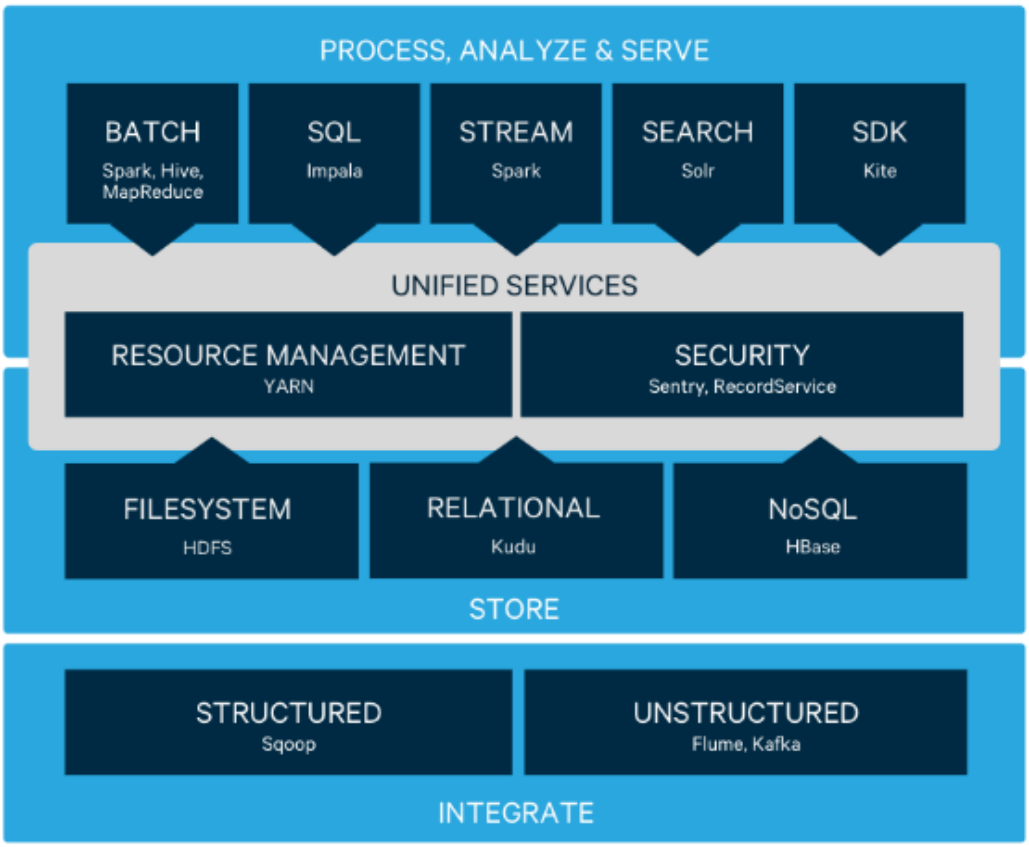
2.11 HDP Stack [36]

Característiques rellevants del stack HDP

- És de codi obert.
- Compatible amb Windows.
- Basat completament en Apache promocionant la creació de noves eines de codi obert.
- Utilitza el gestor/instal·lador Apache Ambari [37] que és molt bàsic.
- Inclou monitoreig proactiu i manteniment amb els membres subscrits.
- Permet encriptació de la informació en repòs.

2.5.2. Stack de Cloudera

Cloudera és la més vella i coneguda distribució de Hadoop. Va ser fundada el 2008 per grans empreses del BigData com Google, Facebook o Oracle. Cloudera ofereix dues distribucions, una *open source Cloudera Distribution for Hadoop* (CDH) i una de codi propietari *Cloudera Managment Suite*. L'objectiu a llarg termini de la companyia és convertir-se en un centre de dades per a les empreses, per reduir la necessitat de magatzems de dades per a les empreses que depenguin d'ella. [36]



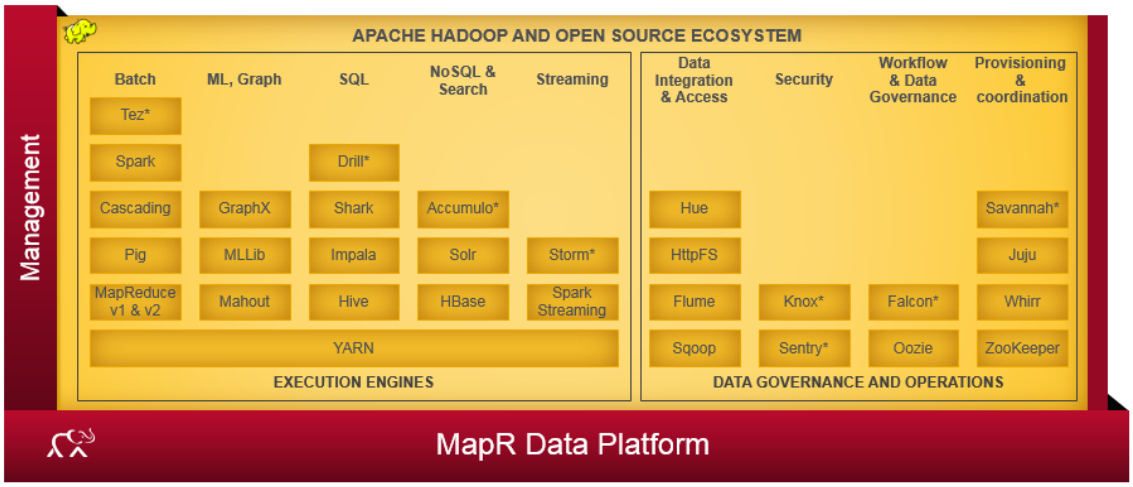
2.12 Cloudera Stack [39]

Característiques rellevants del stack CDH

- És de codi obert.
- Compatible amb Windows Server.
- Interfície amigable per a l'usuari.
- Utilitza el seu propi gestor/instal·lador Cloudera Manager.
- Distribució més lenta que MapR.
- Amplia el nucli de Hadoop amb eines pròpies com Impala o Kudu.
- Conté eines per la gestió operativa i la generació d'informes útils pel món empresarial com la recuperació automàtica de desastres.
- Permet encriptació de la informació en repòs.

2.5.3. Stack de MapR

MapR proveeix una distribució completa de Hadoop a través d'aplicacions no basades en Apache Hadoop fet que la diferencia notablement de les distribucions de Hortonworks i Cloudera. MapR és una solució de codi propietari que basa el seu negoci a partir de millores que fan el sistema més ràpid, confiable i amigable per a l'usuari. [36]



2.13 MapR Stack [40]

Característiques rellevants del stack MapR

- Codi propietari amb versió gratuïta.
- És la distribució més ràpida.
- Interfície de consola dolenta en comparació amb Cloudera.
- Permet una protecció completa de les dades sense punts de fallada.
- Conté altres sistemes com MapR-FS en substitució del HDFS, com també la seva pròpia base de dades NoSQL, MapR-DB.
- Proveeix d'encriptació de la informació enviada a, des de i dins d'un clúster.

	HDP	CDH	MapR
Open Source	100% - all code contributed to Apache, ZERO proprietary	Open Core, focus on proprietary, packagers of open	Rewritten core, no longer Apach e Hadoop, proprietary
Stable Hadoop Version 1	Every release closest to Open S ource Trunk	Fork early patch often – hundred s of proprietary patches	Not Apache, proprietary only
Expertise	19 Hadoop Core 95 Across All Project	8 Hadoop Core ~20 Across All Project	1 Hadoop Core <10 Across All Projects
Focus & Strategic innovat ion	Hadoop 1, Hadoop 2, Ambari, Hive, Ecosystem	Proprietary Components, Impala and Manager	HBase Rewrite HDFS Rewrite
Hadoop 2 –next gen Had oop beyond batch	Architected & Built 95%, only c ompany ready to support	Limited involvement, limited kno wledge of YARN	Packager, unqualified to support
Use Case Fit	Infrastructure enablement and specific use cases	Infrastructure enablement and specific use cases	Online use cases mostly Not well suited for infrastructure
Platform Support	Windows, Linux	Linux Only	Linux Only
QA/Test Readiness	Tested on thousands of node, production ready release only	Limited test environment, release beta as GA	Limited test environment
Ecosystem Enablement & Interoperability	Enable and empower ALL apps to benefit from Hadoop. Focus on app vendors like Microsoft, Teradsata, Splunk	Competitive with Impala, no foc us on Hive. Focus on pull throug h hardware vendors like Dell and HP	Vertical use case focus moves u p stack to competitive. Focus on AWS. Unclear app focus.
Leadership	Invented open source model	Traditional enterprise software	Traditional enterprise software

2.14 Comparativa dels stacks [41]

De la figura 2.14 anterior se'n poden extreure les conclusions que HDP disposa de més compatibilitats gràcies a tenir de codi obert el seu nucli com el total de les aplicacions que utilitza. També evita enfocar-se en promocionar una sola aplicació a diferència dels altres fent més fàcil la nova incorporació d'aplicacions útils. En general la conclusió que se n'obté és que HDP disposa d'una versatilitat inherent en l'ús de codi obert, mentre que les altres distribucions tot i tenir el nucli de codi obert el seu entorn és privatiu o no pru compatible amb les noves aplicacions de codi obert.

Seguint els requeriments abans exposats, s'ha elegit el Stack HDP de Hortonwoks perquè:

- Està basat en l'ideal de l'*open source.*
- Apache el fa servir per a la instal·lació d'Ambari.
- Te una àmplia comunitat i documentació.
- Conté tots els serveis necessaris per a aquest projecte.
- Té compatibilitat amb diferents SO.

3. INSTAL·LACIÓ I CONFIGURACIÓ CLÚSTER BIG DATA

3.1. Requisits Previs

Perquè el sistema funcioni correctament s'han de complir uns requeriments de hardware per a suportar la càrrega de recursos i un software suficientment versàtil per permetre les configuracions necessàries.

- Els nodes hauran de tenir suficients recursos de RAM, CPU i Disc per a poder executar tasques BigData i MPI de forma òptima.
- El SO ha de permetre les configuracions necessàries per a poder interconnectar els nodes del clúster.
- El SO ha de ser compatible amb el stack seleccionat.
- El SO ha de ser compatible amb la instal·lació de MPI.
- El Stack haurà de tenir tots els serveis i aplicacions necessàries per poder dur a terme execucions *MapReduce.*
- El Stack ha de permetre la instal·lació d'Apache Spark.
- L'entorn d'execució ha de permetre aïllament de tasques entre usuaris per garantir-ne la seguretat.

Respecte al SO s'ha elegit CentOS 7 Minimal [42] perquè:

- És el CentOS més recent i per tant gaudirà de més anys de suport.
- És compatible amb el Stack seleccionat.
- Permet aplicar les configuracions requerides per a configurar el clúster.
- Permet instal·lació mínima evitant tenir paquets que no es faran servir.

3.2. Planificació Instal·lació

Per a una correcta instal·lació s'ha d'executar seguint un ordre i planificació que permetin un encaix correcte entre els futurs serveis del clúster i que permeti la correcta coordinació els nodes del sistema.

En aquest cas es disposa de 6 nodes

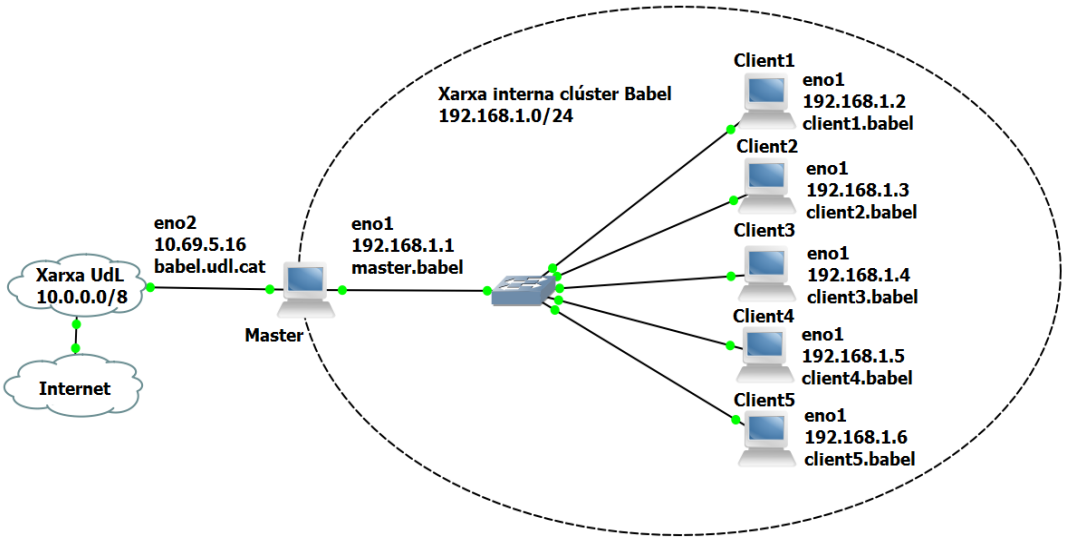
- Un node amb 128GB de RAM @ 2400MHz, 80GB de SSD, 4TB en dos HDD i 2 Intel(R) Xeon(R) CPU E5-2609 v4 @ 1.70GHz de 8 cores.
- Cinc nodes amb 64GB de RAM @ 2400MHz, 80GB de SSD, 4TB en dos HDD i 2 Intel(R) Xeon(R) CPU E5-2609 v4 @ 1.70GHz de 8 cores.

Aquests nodes han estat muntats, instal·lats i configurats per tal de funcionar correctament. En la figura 3.1 s'observa la seva disposició física en el rack.

3.1 Nodes "enrackats"

A partir d'aquestes màquines, s'ha definit que el node amb més RAM serà anomenat master i convertir la resta de nodes en clients.

La distribució dels noms de xarxa/urls seguirà el següent mapa de xarxa de la figura 3.2.



3.2 Xarxa del clúster

Per obtenir una correcta compatibilitat amb Ambari, el sistema solament requereix tenir habilitat l'usuari root. Ambari a l'hora d'instal·lar els diferents serveis s'encarregarà de crear nous usuaris amb els permisos adients per a cada servei.

Com que el node master anirà connectat a la xarxa de la UdL, per evitar problemes solament té accessibles els ports 22, 80 i 445. La interfície web haurà d'anar pel port 80 i no el 8080 com Ambari dictamina per defecte.

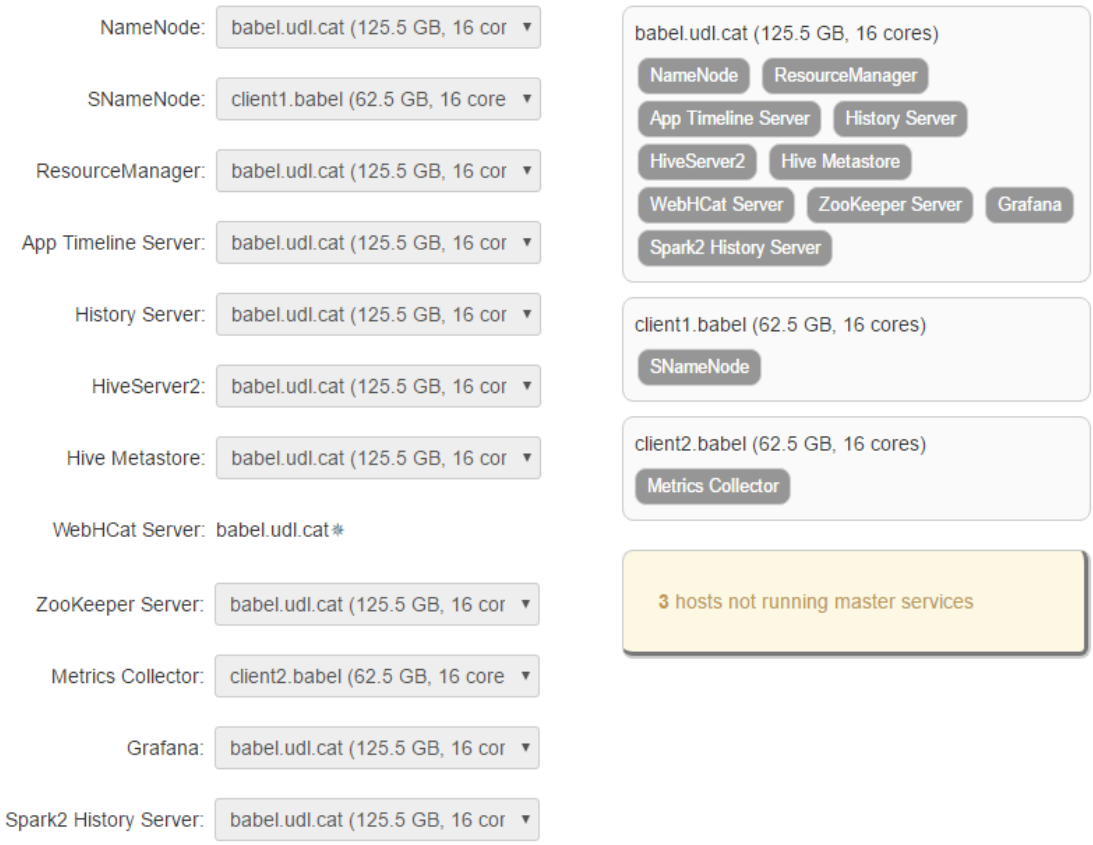
Un cop instal·lat Ambari, s'instal·laran un a un de forma progressiva els serveis necessaris per garantir-ne el correcte acoblament al clúster. Donat el poc temps del qual es disposa de les màquines, s'ha triat una quantitat mínima de serveis a instal·lar, HDFS, MapReduce2, YARN, ZooKeeper, Spark2 (amb les seves dependències Hive, Tez i Pig) i Ambari Metrics.

Donat que cada màquina té 1 SSD i 2 HDD, es reservarà el SSD pel sistema i els HDD per la instal·lació de HDFS, ja que, així serà el mateix sistema de Hadoop l'encarregat de gestionar les dades entre els discos. Si s'utilitzés un RAID entre els discos el rendiment es veuria afectat i el sistema HDFS no podria balancejar correctament la càrrega, en el sistema HDFS és preferible tenir diversos discs/datanodes que un disc molt gran per garantir la correcta replicació de blocs.

3.3.Definir arquitectura

Donada la diversitat de què serveis que disposarà el clúster, s'ha de distribuir de forma correcta els serveis encarregats de la supervisió i gestió del clúster, com també tenir assignats els serveis de dades i execució de treballs en els nodes.

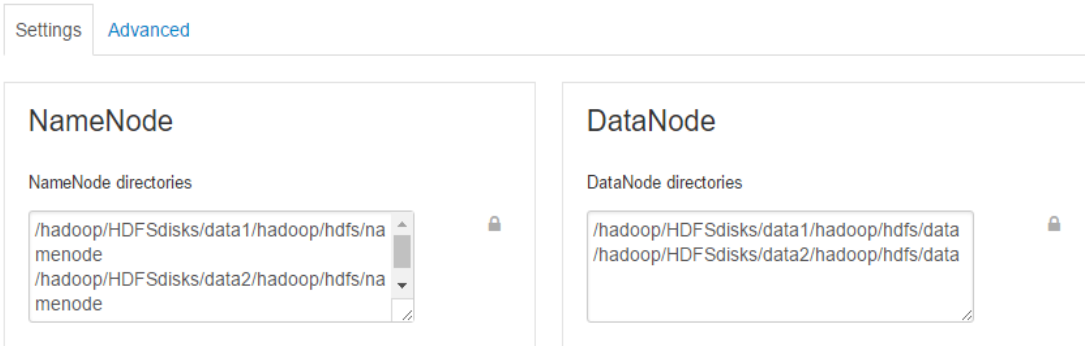
Donat que el node master disposa de més RAM, es concentraran els serveis de gestió i coordinació en aquest. Els nodes client seran els encarregats de tenir els clients dels serveis de treball de HDFS, MapReduce, YARN, ZooKepper, Spark2 (amb les seves dependències Hive, Tez i Pig) i Ambari Metrics. En la figura 3.3 s'observa com queden distribuïts els processos gestors de cada servei del clúster un cop instal·lats. S'ha de tenir en compte el servei encarregat de fer de suport al servei HDFS SNameNode s'ha de ficar en un node diferent del servei NameNode principal per evitar anul·lar el suport a caigudes que ofereix, igual que el servei de Spark Spark2 Thrift Server que Ambari no el mostra en la figura 3.3. També el servei Metrics Collector s'ha ficat en un node diferent de master per raons de connectivitat entre el clúster, ja que la limitació de ports accessibles des del domini babel.udl.cat fa impossible que des de la interfície web que s'accedeix des de fora la xarxa es pogués veure la seva informació, per tant, al ficar un node intern de la xarxa del clúster això queda solucionat.



3.3 Distribució dels processos gestors del clúster

Seguint la lògica anterior dels usuaris, Ambari s'encarrega de gestionar la creació de particions per a cada servei. Generalment, s'instal·laran en */Hadoop*.

Com es compta amb 2 discos extres en cada node, s'afegiran a la configuració com s'observa a la figura 3.4 del HDFS per a tal que el sistema els inclogui.



3.4 Directoris on es muntaran els dos HDD

3.4. Instal·lació y configuració sistema operatiu base

S'han de configurar tots els nodes de la mateixa manera per tal de tenir una correcta configuració apta per Hadoop.

3.4.1. Instal·lació OS i paquets

- Instal·lar CentOS 7 Minimal en el directori / de la partició principal.
- Executar la comanda *yum update* per actualitzar el sistema a l'última versió.
- Executar la comanda *yum install net-tools vim ntp wget* per a instal·lar els paquets necessaris per la configuració del sistema.

3.4.2. Configuració clúster ssh passwordless

- Executar la comanda *ssh-keygen* a la màquina master per a generar els certificats SSH.
- Copiar els certificats generats *.ssh/id_rsa i .ssh/id_rsa.pub* del node master a *.ssh* dels clients.
- Executar les comandes *cat ~/.ssh/id_rsa.pub >> ~/.ssh/*authorized_keys, *chmod 700 ~/.ssh* i *chmod 600 ~/.ssh/authorized_keys* per a obtenir una comunicació entre els nodes SSH sense requeriments de contrasenya.

➢ El parell de claus id_rsa i id_rsa.pub s'han de replicar en un lloc segur també per si mai fan falta.

3.4.3. Configuració de la xarxa

En el node master s'ha d'aplicar encaminament per donar accés a internet a la xarxa interna, s'executaran les següents comandes:

- *echo 1 > /proc/sys/net/ipv4/ip_forward*
- *sysctl -p /etc/sysctl.conf*
- *iptables -t nat -A POSTROUTING -o eno2 -j MASQUERADE,*
- *iptables -A FORWARD -i eno2 -o eno1 -m state --state RELATED,ESTABLISHED -j ACCEPT*
- *iptables -A FORWARD -i eno1 -o eno2 -j ACCEPT*
- *service iptables save*
- *service iptables restart*

A tots els nodes es seguiran els passos següents:

- Executar les comandes *systemctl disable firewalld* i *service firewalld stop* per a desactivar el firewall.
- Per activar el *Network Time Protocol* (NTP) executar les comandes:
  - *systemctl stop chronyd*
  - *systemctl disable chronyd*
  - *systemctl enable ntpd*
  - *systemctl start ntpd*
- Executar la comanda *hostname* <NOM de la màquina> per a establir-li un nom de xarxa.
- Editar el fitxer */etc/sysconfig/network* i deixar els camps amb *NETWORKING=yes HOSTNAME=*<NOM de la màquina>.
- Editar el fitxer */etc/hosts* de tal forma que el sistema tingui coneixença de la resta de nodes amb les seves IP i urls tal com es mostra en la figura *3.5*.

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.1.1 master.babel master
192.168.1.2 client1.babel client1
192.168.1.3 client1.babel client2
192.168.1.4 client1.babel client3
192.168.1.5 client1.babel client4
192.168.1.6 client1.babel client5
```

3.5 **Fitxer /etc/hosts de les màquines**

3.4.4. Configuracions del sistema

- Executar la comanda *setenforce 0* SELINUX.
- Executar les comandes *umask 0022* i *echo umask 0022 >> /etc/profile* per activar establir els permisos per defecte dels nous fitxers que es creïn.
- Per a formatejar i muntar els discos de 2TB als directoris escollits del HDFS s'han de realitzar les següents comandes:
  - *mkdir /hadoop/HDFSdisks*
  - *mkdir /hadoop/HDFSdisks/data1*
  - *mkdir /hadoop/HDFSdisks/data2*
  - *fdisk /dev/sdb* (nova partició primària)
  - *mkfs.ext4 -m 0 /dev/sdb1*
  - *tune2fs -m 0 /dev/sdb1*
  - *fdisk /dev/sdc* (nova partició primària)
  - *mkfs.ext4 -m 0 /dev/sdc1*
  - *tune2fs -m 0 /dev/sdc1*
  - *mount -a*
- Per a que es muntin en cada reinici cal incloure les particions al *fstab* tal com mostra la figura 3.6, caldrà executar les comandes següents:
  - *echo "/dev/sdb1 /hadoop/HDFSdisks/data1 ext4 defaults,noatime 0 0" >> /etc/fstab*
  - *echo "/dev/sdc1 /hadoop/HDFSdisks/data2 ext4 defaults,noatime 0 0" >> /etc/fstab*

```
/dev/mapper/cl-root     /                       xfs     defaults        0 0
UUID=e7725dc3-cf21-4a6f-a8b1-927603875128 /boot                   xfs     defaults        0 0
UUID=E13C-A9EC          /boot/efi               vfat    umask=0077,shortname=winnt 0 0
/dev/mapper/cl-home     /home                   xfs     defaults        0 0
/dev/mapper/cl-swap     swap                    swap    defaults        0 0
/dev/sdb1               /hadoop/HDFSdisks/data1 ext4    defaults,noatime 0 0
/dev/sdc1               /hadoop/HDFSdisks/data2 ext4    defaults,noatime 0 0
```

3.6 Fitxer /etc/fstab de les màquines

3.5. Instal·lació Ambari

3.5.1. Servidor

- Executar la comanda *wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.4.1.0/ambari.repo -O /etc/yum.repos.d/ambari.repo* per a instal·lar el repositori d'Ambari al sistema.
- Executar la comandes *yum install ambari-server -y*, *ambari-server setup -s* per a instal·lar el servidor d'Ambari a la màquina master.
- Editar la configuració de Aambari en */etc/ambari-server/conf/ambari.properties* afegit el camp *client.api.port=80* per tal de fer que la GUI web d'Ambari es vegi pel port obert 80.
- Finalment executar *ambari-server start* per a iniciar el sistema.

3.5.2. Clients

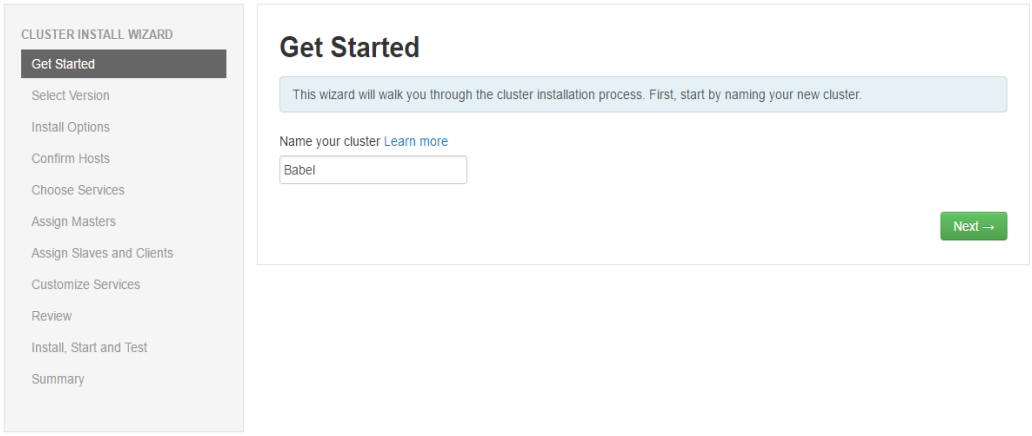
- Executar la comanda *wget -nv http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.4.1.0/ambari.repo -O /etc/yum.repos.d/ambari.repo* per a instal·lar el repositori d'Ambari al sistema.
- Executar la comandes *yum install ambari-client -y* i *ambari-client start* per a instal·lar i iniciar el client d'Ambari a les màquines client.

3.6. Instal·lació Hadoop Stack i configuració dels serveis

Cal tenir en consideració que no es podran utilitzar les UI de les aplicacions, ni altres serveis que requereixin del seu propi port extern perquè no es disposen dels ports lliures necessaris.

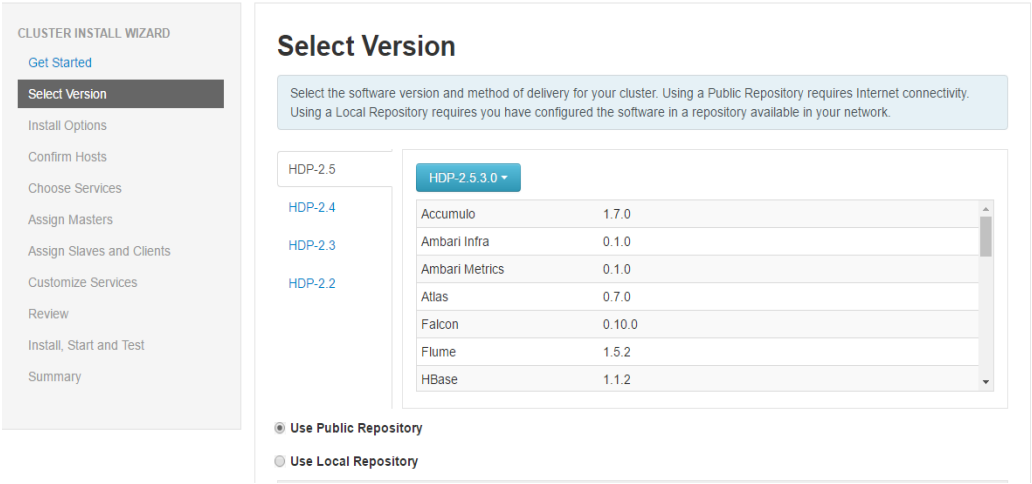
3.6.1. Clúster base

Per tal de poder instal·lar tots els serveis de Hadoop, s'ha d'instal·lar primer la base del clúster, per fer-ho se seguiran els següents passos:



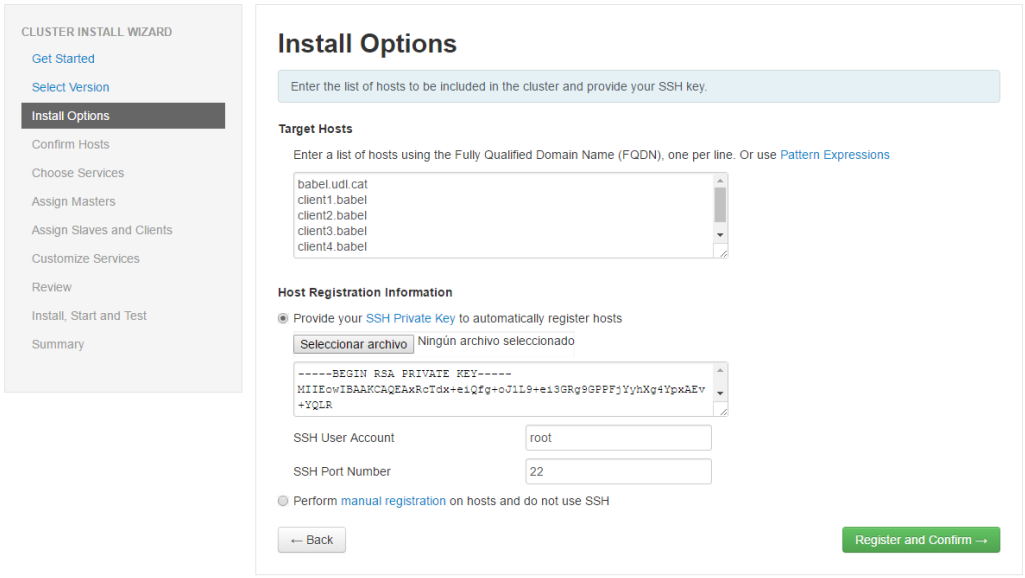
3.7 Pas 1

Seleccionar la versió del Stack a instal·lar, en aquest cas serà l'última.



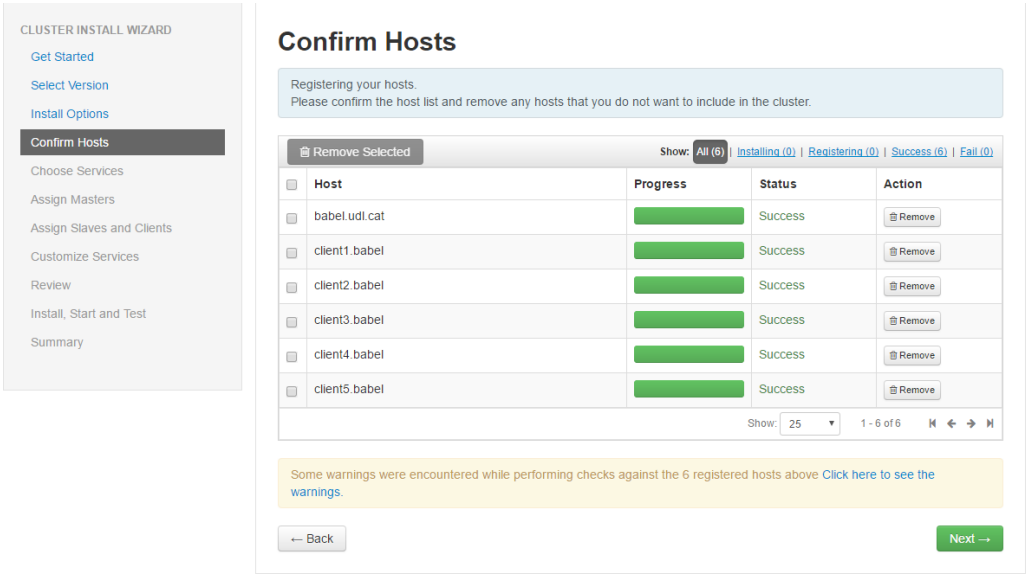
3.8 Pas 2

En el següent pas, tal com s'observa en la figura 3.9, caldrà introduir els noms de domini de les màquines que conformin el clúster. També s'afegirà la clau privada generada anteriorment per permetre la comunicació entre nodes.



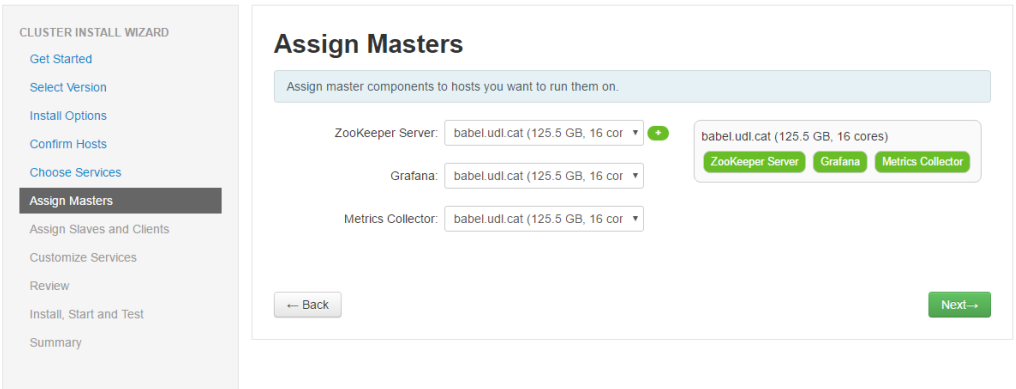
3.9 Pas 3

Seguidament el gestor instal·larà els components inicials a cada node i un cop finalitzat haurà de donar un resultat com el de la figura 3.10. En aquest cas es dóna una alerta per l'ús de *iptables*, però com solament hi ha una política d'encaminament i no es filtra el tràfic no suposarà un problema.



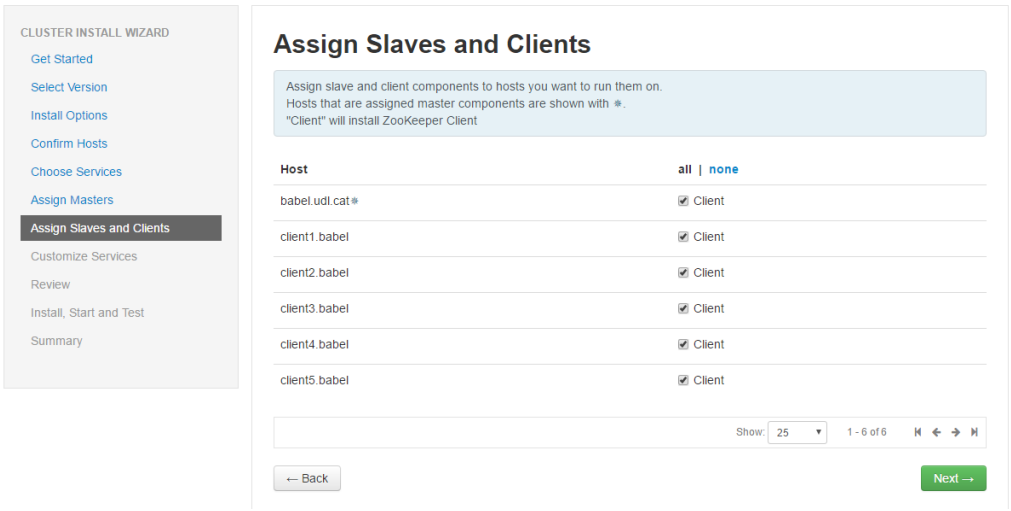
3.10 Pas 4

Un cop completat correctament el pas anterior, s'elegirà perquè el node master sigui l'encarregat de tenir tots els serveis de gestió de les aplicacions bàsiques (Ambari Metrics i ZooKeeper) necessàries per instal·lar el clúster.



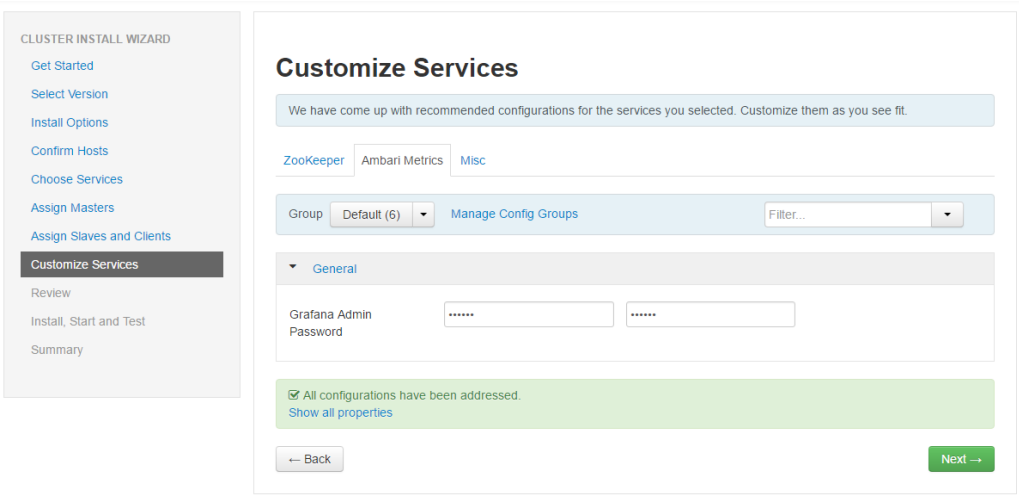
3.11 Pas 5

Com tots els nodes seran de treball, tots ells se'ls instal·larà els clients pertinents dels serveis.



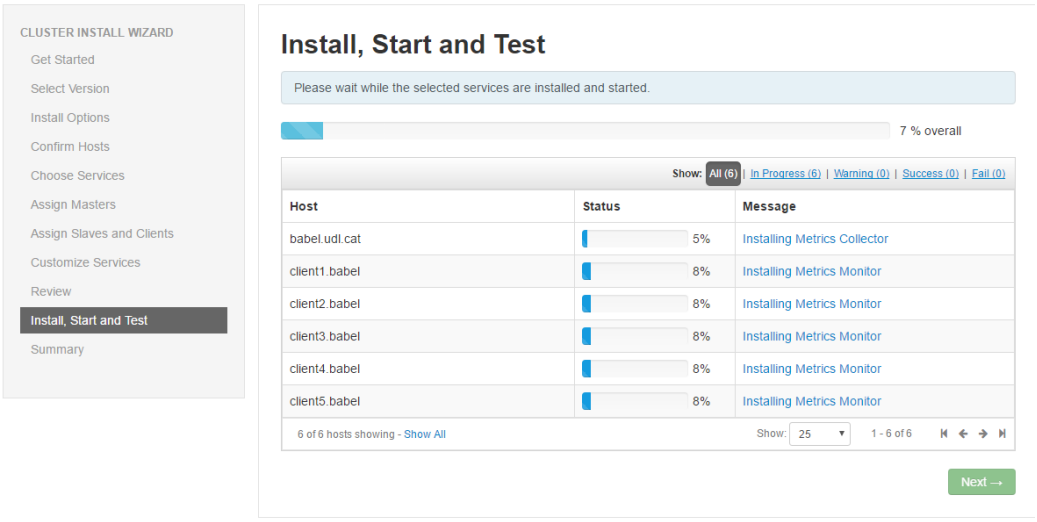
3.12 Pas 6

En el moment de validar la configuració, serà necessari aplicar una contrasenya a un dels serveis d'Ambari Metrics com s'aprecia en la figura 3.13. En el cas mostrat una contrasenya trivial com "123456" serà suficient.

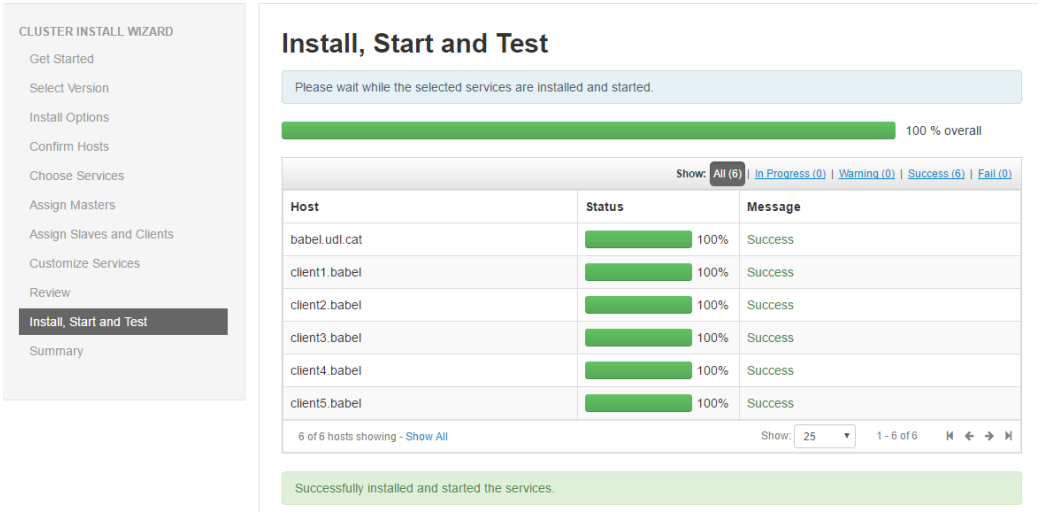


3.13 Pas 7

Finalment es procedeix a desplegar la instal·lació dels serveis. Si tot funciona correctament es passarà d'un estat com el de la figura 3.14, a un estat de finalització correcta com el de la figura 3.15.



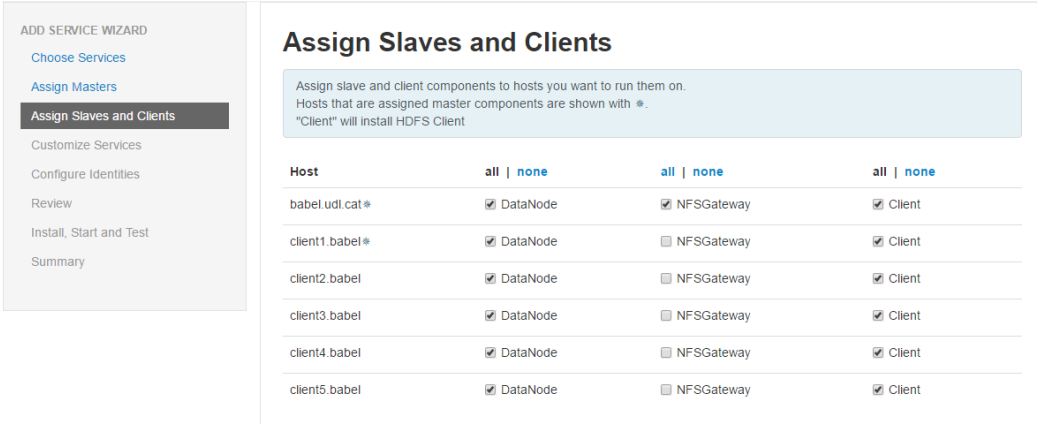
3.14 Pas 8



3.15 Pas 9

3.6.2. HDFS

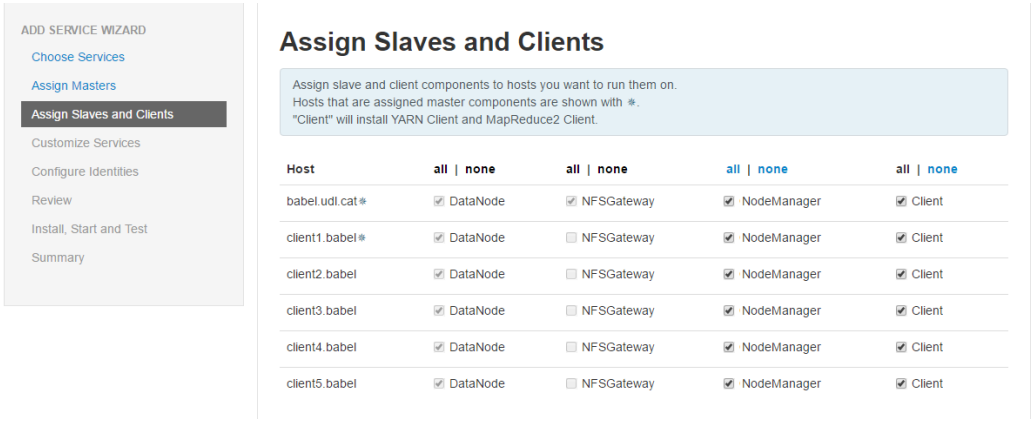
Per a instal·lar el sistema HDFS caldrà anar a l'opció d'afegir un nou servei del gestor Ambari. El procés és similar al de l'apartat anterior, la repartició de clients i serveis de gestió quedarà com el de la figura 3.16. El servei NFSGateway compleix la funcionalitat de permetre als clients muntar el sistema HDFS com a part del seu sistema de fitxers local, com que aquest servei pot anar en qualsevol node del clúster s'ha ficat a master.



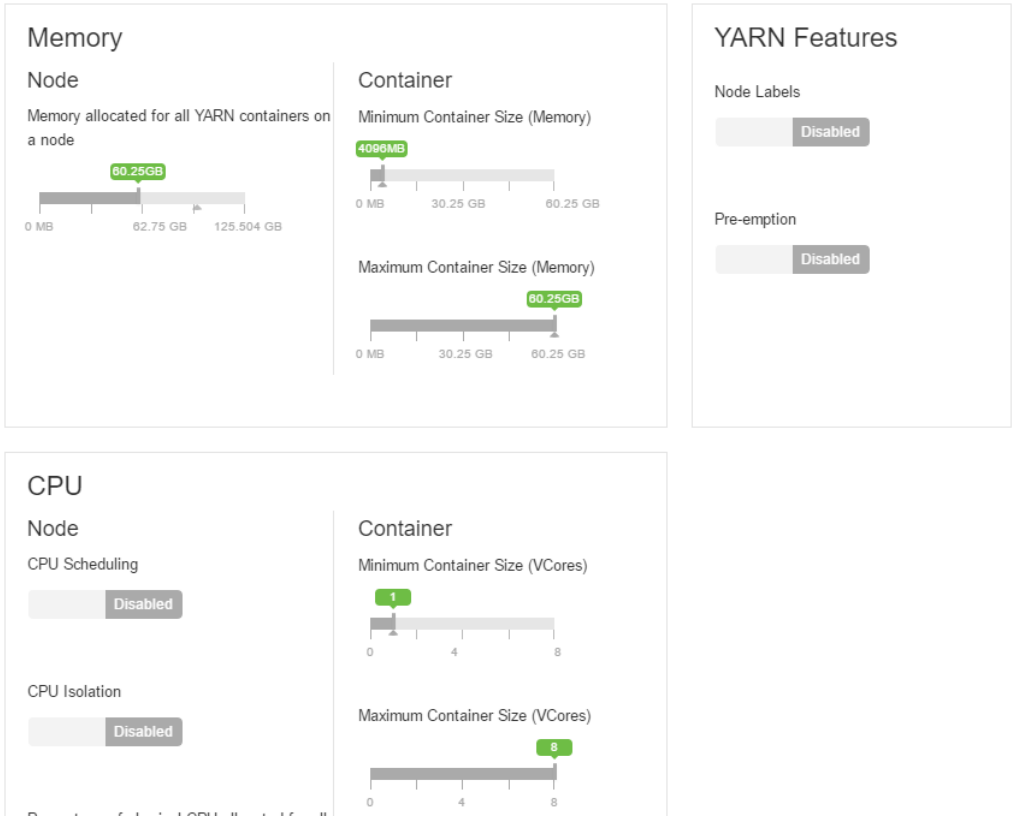
3.16 Distribució de clients i serveis HDFS

3.6.3. YARN MapReduce2

Seguint la retòrica anterior s'instal·larà el YARN aplicant la distribució de clients de la figura 3.17 i la configuració principal ha sigut modificada per tal de ser la correcta pel sistema seguint la figura 3.18.



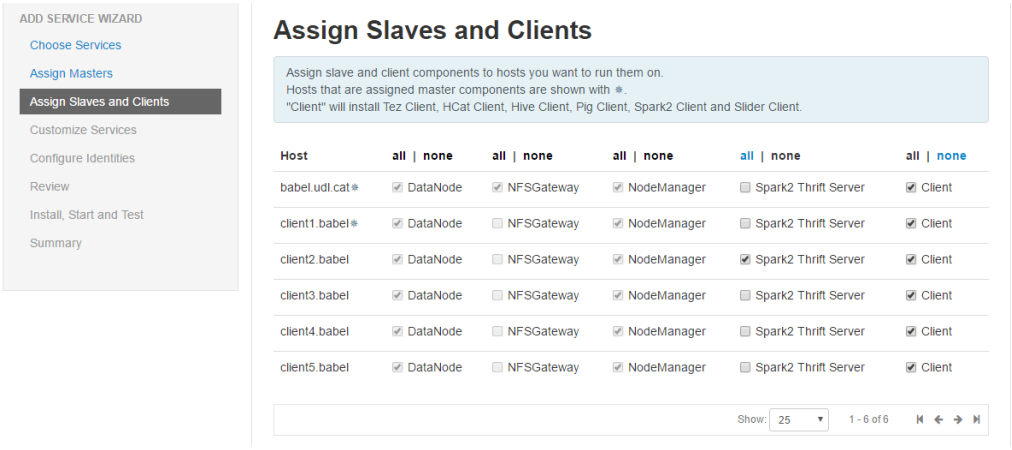
3.17 Distribució de clients i serveis YARN



3.18 Configuració principal YARN

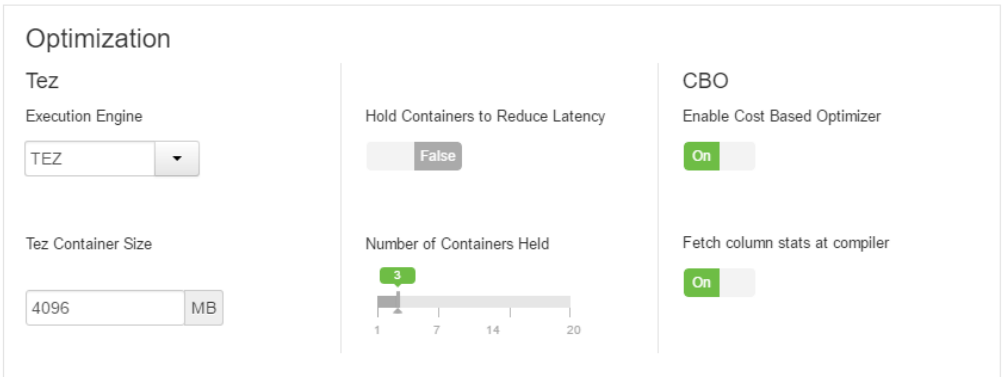
3.6.4. Spark2

Per a poder instal·lar Spark2, el sistema demana que s'instal·lin les seves dependències, Hive, Tez i Pig. S'instal·laran al mateix temps que Spark2 i la distribució de clients i serveis queda com la figura 3.19.



3.19 Distribució de clients i serveis de Spark i les dependències

En aquest cas el sistema demana introduir una contrasenya per la base de dades de TEZ, es farà servir una trivial altre cop "123456". També s'adaptarà la mida de contenidor de Tez seguint indicacions del programa ficant-lo a 4096 MB com mostra la figura 3.20.



3.20 Configuració contenidors TEZ

3.7. Benchmarks BigData per a Hadoop/Spark

S'ha creat un usuari i se li han donat permisos en les carpetes destinades al testing dins el sistema HDFS. Per a fer-ho s'han de seguir uns passos. [43]

Sent l'usuari root en el node master, executar la comanda *adduser -g hadoop testUser* i *passwd testUser* per a crear el l' usuari de test.

Seguidament s'han creat les carpetes necessàries i se li donaran permisos de propietari a l'usuari creat, per a fer-ho s'executaran les comandes següents:

- *sudo -u hdfs hadoop fs -mkdir /benchmarks*
- *sudo -u hdfs hadoop fs -chown -R testUser /benchmarks*
- *sudo -u hdfs hadoop fs -mkdir /user/testUser*
- *sudo -u hdfs hadoop fs -chown -R testUser /user/testUser*

Es comprovarà que la carpeta de testing està correctament assignada amb la comanda *hdfs dfs -getfacl /benchmarks* que donarà un resultat com el de la figura 3.21.

```
[testUser@babel /]$ hdfs dfs -getfacl /benchmarks
# file: /benchmarks
# owner: testUser
# group: hdfs
user::rwx
group::r-x
other::r-x
```

3.21 Permisos carpeta /benchmarks en el sistema HDFS

Per a poder executar els testos elegits [44] s'iniciarà sessió amb l'usuari testUser amb la comanda *su testUser* i s'executarà la comanda *cd /usr/hdp/2.5.3.0-37/hadoop-mapreduce/* per moure's a la carpeta on són els executables per a fer test. Es poden veure les diferents opcions que ens mostra la figura 3.22 amb la comanda *hadoop jar hadoop-*test*.jar*.

```
[testUser@babel hadoop-mapreduce]$ hadoop jar hadoop-*test*.jar
Unknown program 'hadoop-mapreduce-client-jobclient-tests.jar' chosen.
Valid program names are:
  DFSCIOTest: Distributed i/o benchmark of libhdfs.
  DistributedFSCheck: Distributed checkup of the file system consistency.
  JHLogAnalyzer: Job History Log analyzer.
  MRReliabilityTest: A program that tests the reliability of the MR framework by injecting faults/failures
  NNdataGenerator: Generate the data to be used by NNloadGenerator
  NNloadGenerator: Generate load on Namenode using NN loadgenerator run WITHOUT MR
  NNloadGeneratorMR: Generate load on Namenode using NN loadgenerator run as MR job
  NNstructureGenerator: Generate the structure to be used by NNdataGenerator
  SliveTest: HDFS Stress Test and Live Data Verification.
  TestDFSIO: Distributed i/o benchmark.
  fail: a job that always fails
  filebench: Benchmark SequenceFile(Input|Output)Format (block,record compressed and uncompressed), Text(Input|Output)Format (compressed and uncompressed)
  largesorter: Large-Sort tester
  loadgen: Generic map/reduce load generator
  mapredtest: A map/reduce test check.
  minicluster: Single process HDFS and MR cluster.
  mrbench: A map/reduce benchmark that can create many small jobs
  nnbench: A benchmark that stresses the namenode.
  sleep: A job that sleeps at each map and reduce task.
  testbigmapoutput: A map/reduce program that works on a very big non-splittable file and does identity map/reduce
  testfilesystem: A test for FileSystem read/write.
  testmapredsort: A map/reduce program that validates the map-reduce framework's sort.
  testsequencefile: A test for flat files of binary key value pairs.
  testsequencefileinputformat: A test for sequence file input format.
  testtextinputformat: A test for text input format.
  threadedmapbench: A map/reduce benchmark that compares the performance of maps with multiple spills over maps with 1 spill
[testUser@babel hadoop-mapreduce]$
```

3.22 Executables de testing del sistema

3.7.1. TestDFSIO

Per comprovar el correcte funcionament del sistema HDFS hi ha diverses eines per mirar el seu rendiment. En aquest cas utilitzant l'eina TestDFSIO es pot observar les capacitats d'escriptura i lectura del sistema.

S'ha executat un test d'escriptura de 10 fitxers de 1GB amb la comanda *hadoop jar hadoop-mapreduce-client-jobclient-tests.jar TestDFSIO -write -nrFiles 10 -fileSize 1024 -resFile ~/writeTest.txt*. El resultat que mostra la figura 3.23 diu que en uns 92 segons el sistema és capaç d'escriure 10240MB ~ 10GB d'informació, donant una sortida de 16 MB per segon aproximadament.

```
17/06/20 23:15:59 INFO fs.TestDFSIO: ----- TestDFSIO ----- : write
17/06/20 23:15:59 INFO fs.TestDFSIO:            Date & time: Tue Jun 20 23:15:59 CEST 2017
17/06/20 23:15:59 INFO fs.TestDFSIO:        Number of files: 10
17/06/20 23:15:59 INFO fs.TestDFSIO: Total MBytes processed: 10240.0
17/06/20 23:15:59 INFO fs.TestDFSIO:      Throughput mb/sec: 15.958582491377847
17/06/20 23:15:59 INFO fs.TestDFSIO: Average IO rate mb/sec: 16.342309951782227
17/06/20 23:15:59 INFO fs.TestDFSIO:  IO rate std deviation: 2.7443366169665393
17/06/20 23:15:59 INFO fs.TestDFSIO:     Test exec time sec: 92.529
17/06/20 23:15:59 INFO fs.TestDFSIO:
```

3.23 Test d'escriptura del sistema HDFS

El test de lectura s'ha fet de 10 fitxers de 1GB amb la comanda *hadoop jar hadoop-mapreduce-client-jobclient-tests.jar TestDFSIO -read -nrFiles 10 -fileSize 1024 -resFile ~/readTest.txt*. El resultat que mostra la figura 3.24 diu que en uns 79 segons el sistema és capaç de llegir 10240MB ~ 10GB d'informació, donant una lectura de 19.3 MB per segon aproximadament.

```
17/06/20 23:19:12 INFO fs.TestDFSIO: ----- TestDFSIO ----- : read
17/06/20 23:19:12 INFO fs.TestDFSIO:            Date & time: Tue Jun 20 23:19:12 CEST 2017
17/06/20 23:19:12 INFO fs.TestDFSIO:        Number of files: 10
17/06/20 23:19:12 INFO fs.TestDFSIO: Total MBytes processed: 10240.0
17/06/20 23:19:12 INFO fs.TestDFSIO:      Throughput mb/sec: 19.348353500464814
17/06/20 23:19:12 INFO fs.TestDFSIO: Average IO rate mb/sec: 19.624286651611328
17/06/20 23:19:12 INFO fs.TestDFSIO:  IO rate std deviation: 2.4199617699715468
17/06/20 23:19:12 INFO fs.TestDFSIO:     Test exec time sec: 78.734
17/06/20 23:19:12 INFO fs.TestDFSIO:
```

3.24 Test de lectura del sistema HDFS

Com es pot observar, la lectura és més ràpida que l'escriptura, és normal donada la naturalesa d'un disc dur. Tot i que la velocitat no és especialment ràpida si es compara amb un sistema d'un sol disc dur, s'ha de comptar que HDFS per cada bloc que crea ha de replicar-lo arreu del sistema a través de la xarxa, per tant els temps de latència de la xarxa fan baixar bastant el rendiment de E/S.

3.7.2. Terasort

El rendiment conjunt del sistema HDFS + MapReduce el es pot comprovar amb les eines terasort.

Primerament s'executarà la comanda *hadoop jar hadoop-mapreduce-examples.jar teragen 10000000000 /benchmarks/terasort-input* per generar un fitxer de 1TB que serà el que s'ordenarà, en aquest cas segons l'extracte de l'execució figura 3.25 ha tardat 2 hores i 42 minuts en generar el fitxer.

```
StartTime       FinishTime

20-de juny-2017 23:20:56        21-de juny-2017 02:03:32 (2hrs, 42mins, 35sec)
```

3.25 Resultat Teragen

Seguidament executar la comanda *hadoop jar hadoop-mapreduce-examples.jar terasort /benchmarks/terasort-input /benchmarks/terasort-output* que obtindrà el fitxer anterior i l'ordenarà, és el procés que més triga, en aquest cas com demostra la l'extracte de l'execució de la figura 3.26, l'ordenació ha tardat 14 hores i 43 minuts aproximadament.

```
StartTime       FinishTime

21-de juny-2017 02:08:17        21-de juny-2017 03:01:41 (53mins, 23sec)
21-de juny-2017 02:11:24        21-de juny-2017 16:01:51 (13hrs, 50mins, 26sec)
```

3.26 Resultat Terasort

Finalment amb la comanda *hadoop jar hadoop-mapreduce-examples.jar teravalidate /benchmarks/terasort-output /benchmarks/terasort-validate* es valida l'ordenació anterior, seguint l'extracte de l'execució de la figura 3.27 aquest procés ha tardat 3 hores i 17 minuts.

```
StartTime       FinishTime

21-de juny-2017 16:05:02        21-de juny-2017 19:22:29 (3hrs, 17mins, 26sec)
21-de juny-2017 19:22:32        21-de juny-2017 19:22:35 (3sec)
```

3.27 Resultat Teravalidate

Donat el fet que l'ordenació ha estat validada, es pot afirmar que **el clúster permet ordenar 1TB d'informació en 14 hores i 43 minuts**.

3.7.3. HiBench

Actualment existeix una suite de testing a GitHub [45] que combina diferents testos enfocats a comprovar el correcte funcionament del sistema MapReduce i Spark. S'han executat la majoria de testos en el mode "huge" per tal d'obtenir uns rendiments comparatius entre els dos models de processar BigData.

Per a executar els testos s'ha descarregat el repositori git en la carpeta *`/hadoop/testing/HiBench`*. Després s'han adaptat les configuracions *`hadoop.conf`*, *`spark.conf`* i *`hibench.conf`* de la carpeta *`conf`* seguint les indicacions de l'autor. També s'ha modificat el llistat de proves a executar en el fitxer *`benchmarks.lst`* excloent proves que podrien donar problemes pel seu temps d'execució. El llistat final de proves executades es pot observar a la figura X. Una vegada s'ha configurat tot bé, per a executar-ho s'ha anat a la carpeta *`bin`* i executat l'script *`run_all.sh`* que s'encarregarà a partir de la llista anterior l'execució de totes les proves a partir de les configuracions anteriorment generades.

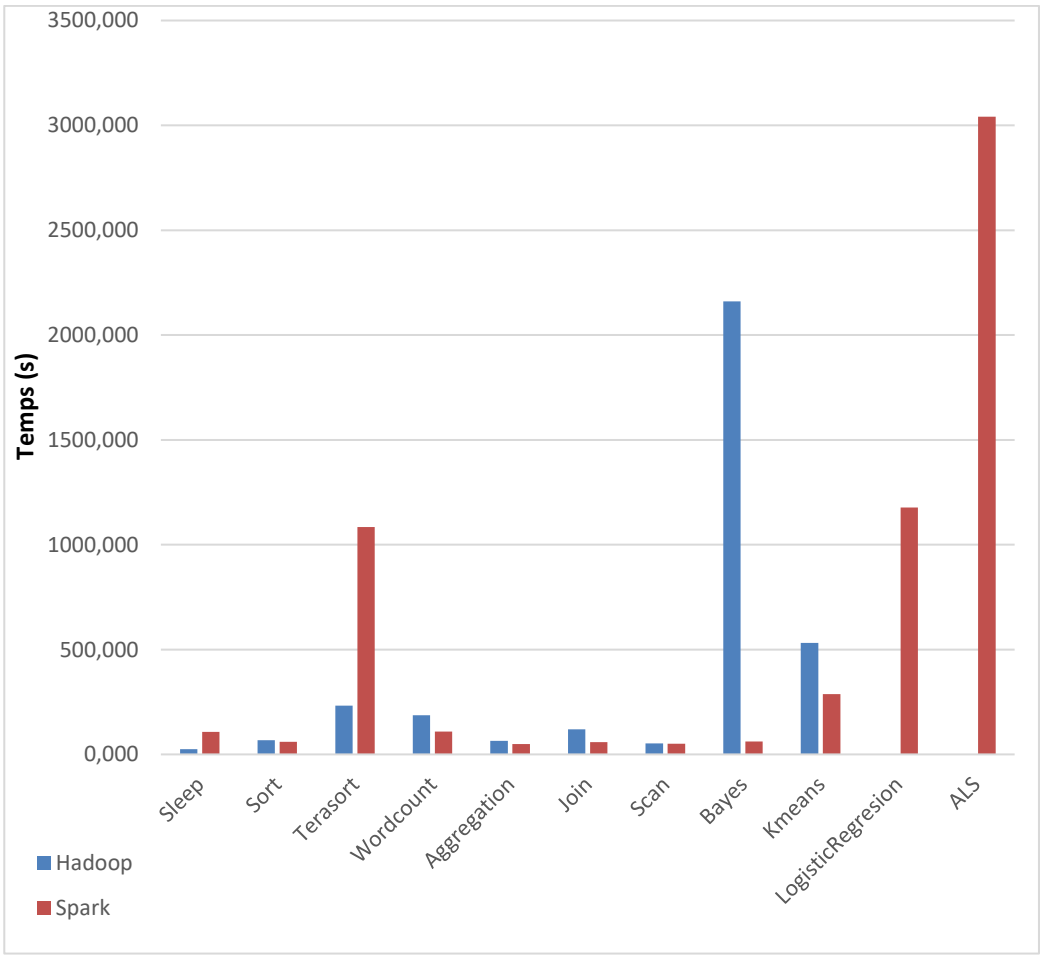
Cal tenir en compte que en aquest cas donat el fet que l'execució és generada des de consola i no des de l'entorn Hadoop i els resultats es guarden en local a l'equip, l'usuari fet servir és root, ja que de fer servir un altre les connexions SSH entre els equips no funcionarien bé perquè només té accés sense contrasenya entre ells des de root.

Tal com s'observa en les figures 3.28 i 3.29 en termes de SQL i de Machine Learning Spark te un major rendiment que el MapReduce de Hadoop, tot i així, en els casos en què s'utilitza molt el sistema HDFS Spark està en desavantatge, ja que la seva utilitat resideix en les operacions en memòria, no en les operacions de disc on el MapReduce al ser de més baix nivell pot anar més ràpid.

Les diferencies més grans es trobem en l'ús de terasort que en utilitzar molt HDFS fa que la utilització de Spark no surti rentable, envers l'algoritme Bayes en què en ser primordialment càlcul Spark obté un gran rendiment en comparació a Hadoop.

Tipus	Nom	Duració (s)	Bytes d'entrada	Sortida (bytes/s)	Sortida/node
Micro Benchmarks	HadoopSleep	24,804	0	0	0
	ScalaSparkSleep	106,942	0	0	0
	HadoopSort	67,456	3284945999	48697610	8116268
	ScalaSparkSort	60,721	3284945999	54099010	9016501
	HadoopTerasort	233,293	32000000000	137166567	22861094
	ScalaSparkTerasort	1083,591	32000000000	29531437	4921906
	HadoopWordcount	186,634	32849105332	176008151	29334691
	ScalaSparkWordcount	108,676	32849105332	302266418	50377736
SQL	HadoopAggregation	64,729	372382246	5752942	958823
	ScalaSparkAggregation	50,010	372382246	7446072	1241012
	HadoopJoin	118,895	1919260193	16142480	2690413
	ScalaSparkJoin	58,894	1919260193	32588382	5431397
	HadoopScan	52,534	2009577838	38252899	6375483
	ScalaSparkScan	50,475	2009577838	39813330	6635555
Machine Learning	HadoopBayes	2160,114	1881786739	871151	145191
	ScalaSparkBayes	62,065	1881786739	30319612	5053268
	HadoopKmeans	532,419	20081826208	37718087	6286347
	ScalaSparkKmeans	287,294	20081826208	69899915	11649985
	LogisticRegression	1176,763	24000602600	20395442	3399240
	ALS	3041,430	9601005600	3156740	526123

3.28 Resultat testos HiBench



3.29 Gràfica de temps dels testos HiBench

4. CONLUSIONS

4.1. Línies d'actuació futures

Donada la manca de temps no s'han pogut assolir part dels objectius plantejats inicialment en aquest projecte, ha sigut així pel fet que no s'ha disposat de les màquines físiques fins a les últimes setmanes d'aquest projecte. També, gràcies al fet de desenvolupar-lo s'han observat noves línies d'actuació que podrien donar més versatilitat al sistema.

Primerament s'hauria d'obrir més ports per a poder accedir a les diferents interfícies gràfiques que dóna el sistema, per així, tenir un millor control dels resultats, perquè ports 22, 80 i 445 només permeten les connexions SSH i la visualització de l'entorn web d'Ambari.

Les principals línies a seguir immediatament després del desenvolupament fet serien les destinades a obtenir una configuració òptima pel sistema, ja que, amb testos fets s'observa que generalment tot funciona correctament, però són testos genèrics, així que s'hauria d'anar iterant diferents testos adaptats a les necessitats del projecte i anar perfilant correctament les configuracions dels serveis.

Seguidament s'hauria d'establir un sistema d'usuaris seguint el mateix esquelet UNIX de permisos que inclou HDFS, en un principi és suficient per al projecte actual, però seria correcte estudiar-ne millores o l'ús de serveis especialitzats en gestió de la seguretat com Kerberos.

Estabilitzades les línies anteriors de millora comentades, el següent pas adient podria passar per establir un sistema centralitzat d'accés als recursos del clúster, ja que, aprofitant l'arquitectura de la xarxa que té un node master que fa de connexió amb "el món", per aplicar-ho sembla que fent servir el servei Apache Mesos seria suficient.

Altrament per tal d'assolit l'objectiu d'un clúster híbrid, es podria afegir un sistema de scripts per tal d'aconseguit fer funcionar els dos sistemes comentats en el plantejament del projecte, Hadoop i MPI. Fent això en teoria s'aconseguiria d'una forma senzilla aquesta combinació tot i que, el pas següent de refinament d'això depèn d'intentar incloure MPI al gestor de recursos YARN simulant més o menys el que fa SGE.

Finalment, una altra bona millora seria la utilització del sistema HUE per a oferir una millor interfície d'interacció perquè, tot i que Ambari és potent en la gestió de serveis, és un gestor global bastant senzill.

4.2. Opinió personal

Personalment penso que aquest projecte té molt més potencial del que se li ha pogut treure, principalment per problemes amb la disponibilitat de l'equipament, amb més temps hauria pogut fer una experimentació i validació més amplia del sistema millorant-ne l'eficiència en general. Tot i així amb el temps del qual he disposat he pogut gaudir d'aprendre el que és muntar un clúster real des de 0 fins a fer-lo funcionar. Certament pels motius abans exposats el clúster no queda en les condicions plantejades inicialment en el projecte, ni tampoc del tot llest, ja que falta perfilar-lo, però en general l'experiència ha sigut gratificant des del punt de vista dels resultats, ja que, tot funciona correctament i afegir noves eines no és un complicat gràcies al gestor Ambari. Personalment el fet de començar de 0 i fer que tot aquest aglomerat de hardware i software funcionin, venint d'algú pràcticament nou en aquest món ja és una fita per la qual em sento orgullós.

També he pogut aprendre bastant sobre el món BigData i gràcies a l'estudi inicial fet conèixer moltes eines i distribucions, això m'ha fet adonar del fet que les tecnologies d'aquest camp ja començant a créixer de forma exponencial, només cal veure la de moviment a GitHub que hi ha al respecte.

Per acabar val a dir que el projecte en si ha sigut prou engrescador, i que, sempre que pugi intentaré seguir investigant per aquest ja no tan nou món del BigData, i com ja vaig fer en altres moments, col·laborar amb la UdL i/o el professorat per poder seguir formant-me en l'àmbit.

5. BIBLIOGRAFÍA

1. Wikipedia, Big Data, https://es.wikipedia.org/wiki/Big_data, 2017
2. John Poppelaars, Is Big Data Objective, Truthful and Credible?, http://john-poppelaars.blogspot.com.es/015/01/is-big-data-objective-truthful-and.html, 2015
3. Dan Shewan, The Internet of Things Is Already Here – and There's Nothing You Can Do About It, http://www.wordstream.com/blog/ws/2015/01/09/the-internet-of-things, 2015
4. SQL Authority, Big Data – What is Big Data – 3 Vs of Big Data – Volume, Velocity and Variety – Day 2 of 21, https://blog.sqlauthority.com/2013/10/02/big-data-what-is-big-data-3-vs-of-big-data-volume-velocity-and-variety-day-2-of-21/, 2013
5. Anil Jain, 5 Vs BigData, https://www.ibm.com/blogs/watson-health/the-5-vs-of-big-data/, 2016
6. Wikipedia, Clúster, https://es.wikipedia.org/wiki/Cl%C3%BAster_(inform%C3%A1tica), 2017
7. Wikipedia, Hadoop, https://es.wikipedia.org/wiki/Hadoop, 2017
8. Dean, J. and Ghemawat, S., 2008. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, *51*(1), pp.107-113, 2008
9. Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." In *ACM SIGOPS operating systems review*, vol. 37, no. 5, pp. 29-43. ACM, 2003.
10. Mubeen Khalid, Hadoop Tutorial, https://www.codegravity.com/blog/hadoop-tutorial, 2015
11. Apache, Spark,http://spark.apache.org/, 2017
12. Harald van der Weel, Hadoop Ecosystem 2015, https://www.linkedin.com/pulse/hadoop-ecosystem-2015-harald-van-der-weel, 2015
13. Microsoft, Hight-Availability System Architecture, https://msdn.microsoft.com/en-us/library/cc750543.aspx, 2002
14. MIT, StarCluster Guides, http://star.mit.edu/cluster/docs/0.93.3/guides/sge.html, 2011
15. Open MPI, Open MPI, https://www.open-mpi.org/, 2017
16. Wisconsin University, HTCondor, https://research.cs.wisc.edu/htcondor/, 2017.
17. Berkeley University, BOINC, https://boinc.berkeley.edu/, 2017
18. Glennklockwood.com, Conceptual _Overview of Map-Reduce and Hadoop, http://www.glennklockwood.com/data-intensive/hadoop/overview.html#3-1-the-magic-of-hdfs, 2015
19. Apache, Apache Hadoop YARN, https://hadoop.apache.org/docs/stable/hadoop-yarn/hadoop-yarn-site/YARN.html, 2016
20. Cloudera, Introduction to YARN and MapReduce 2, https://es.slideshare.net/cloudera/introduction-to-yarn-and-mapreduce-2, 2013
21. Shubham Sinha, Hadoop Ecosystem: Hadoop Tools for Crunching Big Data, https://www.edureka.co/blog/hadoop-ecosystem, 2016
22. Wikipedia, Apache Storm, https://en.wikipedia.org/wiki/Storm_(event_processor), 2017

23. Wikipedia, Apache Flink, https://en.wikipedia.org/wiki/Apache_Flink, 2017
24. Apache, Apache Tez, https://tez.apache.org/, 2017
25. Apache, Apache Crunch, https://crunch.apache.org/, 2013
26. Apache, Welcome to Apache Giraph, http://giraph.apache.org/, 2016
27. Wikipedia, Apache Kafka, https://es.wikipedia.org/wiki/Apache_Kafka, 2017
28. Wikipedia, Apache Avro, https://en.wikipedia.org/wiki/Apache_Avro, 2017
29. Apache, Apache Thrift, https://thrift.apache.org/, 2017
30. Wikipedia, CouchDB, https://es.wikipedia.org/wiki/CouchDB, 2016
31. Wikipedia, Apache Cassandra, https://es.wikipedia.org/wiki/Apache_Cassandra, 2017
32. Wikipedia, Bigtable, https://en.wikipedia.org/wiki/Bigtable, 2017
33. Wikpedia, Apache Accumulo, https://en.wikipedia.org/wiki/Apache_Accumulo, 2017
34. Wikipedia, Apache Mesos, https://es.wikipedia.org/wiki/Apache_Mesos, 2017
35. Xplenty, Cascading, http://www.cascading.org/, 2017
36. American Digital, Cloudera, Hortonworks, and MapR: Comparing the top three Hadoop distributions, http://www.americandigital.com/cloudera-hortonworks-and-mapr-comparing-the-top-three-hadoop-distributions/, 2015
37. Apache Ambari, Apache, https://ambari.apache.org/, 2017
38. Hortonworks, HDP, https://es.hortonworks.com/products/data-center/hdp/, 2017
39. Cloudera, Apache Hadoop open source, https://www.cloudera.com/products/open-source/apache-hadoop.html, 2017
40. MapR, MapR sandbox for Hadoop, https://mapr.com/products/mapr-sandbox-hadoop/, 2017
41. Hortonworks vs Cloudera vs MapR, Mungeol Heo, http://mungeol-heo.blogspot.com.es/2015/01/hortonworks-vs-cloudera-vs-mapr.html, 2015
42. CentOS Project, CentOS, https://www.centos.org/, 2017
43. Setting Up Multiple Users in Hadoop Clusters, https://amalgjose.com/2013/02/09/setting-up-multiple-users-in-hadoop-clusters, 2013
44. Benchmarking a Hadoop Cluster, ace-subido, https://gist.github.com/ace-subido/0a9b219b2348921f6a87, 2017
45. HiBench, michaelmior, https://github.com/intel-hadoop/HiBench, 2017
46. MINF, UdL, http://www.masterinformatica.udl.cat/en/pla-formatiu/objectius-competencies.html, 2017